NON-ADJOINT SURFACTANT FLOOD OPTIMIZATION OF NET PRESENT VALUE AND INCORPORATION OF OPTIMAL SOLUTION UNDER GEOLOGICAL AND ECONOMIC UNCERTAINTY

A Thesis

by

UCHENNA O. ODI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2009

Major Subject: Petroleum Engineering

NON-ADJOINT SURFACTANT FLOOD OPTIMIZATION OF NET PRESENT VALUE AND INCORPORATION OF OPTIMAL SOLUTION UNDER GEOLOGICAL AND ECONOMIC UNCERTAINTY

A Thesis

by

UCHENNA O. ODI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Robert H. Lane
Committee Members,	Maria A. Barrufet
	Yuefeng Sun
Head of Department,	Stephen A. Holditch

December 2009

Major Subject: Petroleum Engineering

ABSTRACT

Non-adjoint Surfactant Flood Optimization of Net Present Value and Incorporation of Optimal Solution Under Geological and Economic Uncertainty. (December 2009)

Uchenna O. Odi, B.S., University of Oklahoma

Chair of Advisory Committee: Dr. Robert H. Lane

The advent of smart well technology, which is the use of down hole sensors to adjust well controls (i.e. injection rate, bottomhole pressure, etc.), has allowed the possibility to control a field in all stages of the production. This possibility holds great promise in better managing enhanced oil recovery (EOR) processes, especially in terms of applying optimization techniques. However, some procedures for optimizing EOR processes are not based on the physics of the process, which may lead to erroneous results. In addition, optimization of EOR processes can be difficult, and limited, if there is no access to the simulator code for computation of the adjoints used for optimization.

This research describes the development of a general procedure for designing an initial starting point for a surfactant flood optimization. The method does not rely on a simulator's adjoint computation or on external computing of adjoints for optimization. The reservoir simulator used for this research was Schlumberger's Eclipse 100, and optimization was accomplished through use of a program written in Matlab. Utility of the approach is demonstrated by using it to optimize the process net present value (NPV) of a 5-spot surfactant flood (320-acres) and incorporating the optimization solution into

a probabilistic geological and economic setting. This thesis includes a general procedure for optimizing a surfactant flood and provides groundwork for optimizing other EOR techniques.

This research is useful because it takes the optimal solution and calculates a probability of success for possible NPVs. This is very important when accessing risk in a business scenario, because projects that have unknown probability of success are most likely to be abandoned as uneconomic. This thesis also illustrates possible NPVs if the optimal solution was used.

DEDICATION

For my family and to God in Jesus Christ

ACKNOWLEDGEMENTS

I would like to thank Dr. Lane for providing me with the Matlab license and graduate research funding for this research. I would like to thank Dr. Jafarpour for the hard data of the porosity field used in this research. I would also like to thank the guidance of Dr. Barrufet, Dr. Sun, and my colleagues in the Department of Petroleum Engineering here at Texas A&M University.

TABLE OF CONTENTS

Page

Page

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

Page

Page

Page

Page

Page

Page

Page

1. INTRODUCTION

Economic and geological uncertainty play a critical role in deciding the producing life of an oil field. Although there may still be producible oil in a field, economics of water handling and geological uncertainty of permeability fields and to a lesser extent porosity fields make producing the remaining oil economically very arduous. Currently, early water breakthroughs (which exacerbate the economics) limit oil recovery for carbonate reservoirs to less than 10% or 25%, while for sandstone reservoirs it ranges from 10% to 35% (Sun, 2009). Typically, after primary production a waterflood is employed for pressure support and to further sweep any oil that is left over; thus decreasing the field oil saturation. One potential drawback, though, for the waterflood is the inability to accurately sweep the oil due to an inadequate mobility ratio. Because of this drawback, it is important to investigate an alternative to a waterflood that does not rely on the mobility ratio alone to effectively displace any oil. One example of such is the surfactant flood. Surfactant floods differ from water floods in that surfactant floods rely on reducing the interfacial tension between the displacing fluid and displaced fluid (Barrufet, 2008b). A variant of the Ensemble Kalman filter (Enkf) was used as the optimization method in this research. This optimization approach is not a direct corollary to Enkf because it does not match a set of observations, but rather uses Enkf to calculate the gradient needed for optimization.

This thesis follows the style of *SPE Journal*.

2. LITERATURE REVIEW

Literature reviews for this research involved studying the past history of surfactant flood optimization and water flood optimization. Each area was unique in its perspective of optimization and helped in gaining an understanding of how better to optimize the surfactant flood process. In order to understand some of the terminology in the literature review, as well as the approach used to develop the product of this research, it is important to review the basic adjoint theory, Kalman Filter, Ensemble Kalman Filter, and the optimization method used in this work.

2.1 ADJOINT THEORY

An adjoint in optimization terminology is generally used to compute a gradient for an optimization process (Zandviet, et al., 2007). To use the adjoint method for calculation of a gradient one must first define an objective function S which represents the NPV. For this explanation of adjoints the objective function to be used is for the optimal control problem in which there is a desire to find a set of controls that maximizes the NPV for a horizontal well waterflood. The reservoir simulator for this process is discrete-time dynamic system model which can be represented by the following equation (Brouwer, et al., 2002).

$$z[m(v+1), m(v), x(v)] = 0 \qquad \textbf{(1)}$$

"z" is a nonlinear function, m is the vector representing the model variables in each grid block (pressures, phase saturations, etc.) of the reservoir simulator, v represents the timesteps from v=0 to v=V, and x represents the vector of control variables for the simulator (injection rates, bottomhole pressures, etc.). In order to perform an optimization the change in the objective function S with respect to x must be accessed. This is very hard to do because x and S are a function of the dynamic system m. To fix this, workers have introduced a Lagrangian that summarizes the dynamic system represented by the reservoir simulator and incorporates the objective function (Brouwer, et al., 2002).

$$L[m(v+1), m(v), x(v), \lambda_a(v+1)] = S[m(v), x(v)] + \lambda_a(v+1)^T z[m(v+1), m(v), x(v)] \tag{2}$$

The λ_a is referred to as the vector of Lagrange multipliers. The Lagrange multipliers can be found by solving a system of adjoint equations.

The adjoint equation (Brouwer, et al., 2002):

$$\lambda_a(v)^T = \left[-\frac{\partial S(v)}{\partial m(v)} - \lambda_a(v+1)^T \frac{\partial z(v)}{\partial m(v)} \right] \left[\frac{\partial z(v-1)}{\partial m(v)} \right]^{-1} \tag{3}$$

Final condition for the adjoint equation

$$\lambda_a(v)^T = 0^T \tag{4}$$

To find the optimal control x consists of doing the following (Brouwer, et al., 2002).

1. Running the reservoir simulator from timestep 0 to V using an initial guess of the control x.
2. Solving the adjoint system backwards through time from timestep V to 0.
3. Compute the gradients of the Lagrangian with respect to the controls x.

$$\frac{\partial L(v)}{\partial x(v)} = \lambda_a(v)^T \frac{\partial z(v)}{\partial x(v)} + \frac{\partial S(v)}{\partial x(v)} \tag{5}$$

4. Updating the controls x (ε is a weighting factor).

$$x(v)_{new} = x(v)_{old} + \varepsilon \frac{\partial L(v)}{\partial x(v)} \quad \textbf{(6)}$$

5. Repeating steps 1 through 4 until there is no change in the Lagrangian gradient.

Observing the steps in using the adjoint solution for the optimization of an objective function, one can see that the solution requires the use of a simulator that allows the backward integration of the dynamic system. This implies that the adjoint solution for optimization relies solely on the dynamic system or the reservoir simulator being used. If the reservoir simulator does not have the option of giving the adjoints through time for the calculation of the Lagrangian gradients then the adjoint solution for optimization cannot be used. Therefore, there is an obvious disadvantage if the simulator does not give the adjoints through time.

2.2 KALMAN FILTER THEORY

Kalman Filter is a method that has been employed where there is a desire to incorporate a set of observations of a system to update the model variables which define the same system. In reservoir engineering Kalman filter is used to history match. Kalman Filter is used as a history match tool when there is an existing reservoir model and a new set of observations (such as water cut) that need to be incorporated to update the model. The Kalman Filter can then be used to update the model variables that define the system such as the pressures, phase saturations, and other geological properties (Gu, et al., 2005). The Kalman Filter consists of two primary steps which are the forecast step (moving the system forward in time) and the assimilation step which the variables defining the system are updated to honor the set of observations (Gu, et al., 2005). In reservoir engineering the forecast step is accomplished by using a reservoir simulator (Gu, et al., 2005). To use Kalman Filter the state vector, y, first has to be defined. This state vector contains the two parts which are the variables that define the system (pressures, phase saturations, and other geological properties) and the theoretical data (water cut, production rates, bottomhole pressures, and etc.). Mathematically the state vector can be written as the following expression (Gu, et al., 2005):

$$y = \left[m^T d^T\right]^T \tag{7}$$

where *m* is a vector that contains the model variables and *d* is a vector containing the simulated data. The best estimate of y that honors the data can be found by minimizing the history match objective function (Gu, et al., 2005):

$$S(y) = \frac{1}{2}\left(My - d_{obs}\right)^T C_D^{-1}\left(My - d_{obs}\right) + \frac{1}{2}\left(y - y_p\right)^T C_Y^{-1}\left(y - y_p\right) \tag{8}$$

where M is a matrix represent by M=[0|I]. 0 is a N_d by N_y-N_d matrix where N_d is the number of measurements and N_y is the number of variables in the state vector. I is a matrix of consisting of 1s with dimensions N_d by N_d. C_D is matrix represent that represents the covariance of the data noise. C_Y is a matrix that represents the covariance of the state vector y while y_p is the prior estimate of y. If one assumes that the relationship between the model variables and the data is linear (d=Gm where G is the sensitivity matrix) C_Y can be written as the following (Gu, et al., 2005):

$$C_Y = \begin{bmatrix} C_m & C_m G^T \\ GC_m & GC_m G^T \end{bmatrix} \tag{9}$$

where C_m is the covariance matrix of model variables.

The estimate of y found by minimizing the objective function is the following (Gu, et al., (2005):

$$y_u = y_p + C_Y M^T \left(M C_Y M^T + C_D \right)^{-1} \left(d_{obs} - M y_p \right) \tag{10}$$

Or in terms of of C_m and G,

$$m_u = m_p + C_m G^T \left(G C_m G^T + C_D \right)^{-1} \left(d_{obs} - M y_p \right) \tag{11}$$

Note though that

$$C_Y M^T = \begin{bmatrix} C_m G^T \\ G C_m G^T \end{bmatrix} \tag{12}$$

and

$$M C_Y M^T = G C_m G^T \tag{13}$$

This implies the covariance of the state variable can be related to the sensitivity matrix and model covariance. This is an important observation in adjoint and non-adjoint based optimization problems because one only needs a representation of the covariance matrix of the state variable *y* to approximate the sensitivity matrix.

2.3 ENSEMBLE KALMAN FILTER THEORY

The Ensemble Kalman Filter is a variant of the Kalman Filter method. The main difference however is that the covariance matrix, C_Y, of the state vector, *y*, is calculated using an ensemble of state vectors representing different realizations of *y* (Gu, et al., 2005). The initial ensemble of state vectors are created by sampling from a probability distribution that represents the variables in the state vector. The ensemble is then updated using the ensemble approximation of the Kalman gain matrix where the Kalman gain matrix is written as the following (Gu, et al., 2005):

$$K_e = C_{Y,e} M^T \left(M C_{Y,e} M^T + C_D \right)^{-1} \tag{14}$$

The updated state vector for the i^{th} realization (total of Ne realizations) is then

$$y_{u,i} = y_{p,i} + K_e \left(d_i - M y_{p,i} \right) \tag{15}$$

The covariance of the state variables can be calculated from the ensemble using following covariance formula (Gu, et al., 2005)

$$C_{Y,e} = \frac{1}{Ne-1} \left(Y_p - \overline{Y}_p \right) \left(Y_p - \overline{Y}_p \right)^T \tag{16}$$

where Y is a matrix represent the ensemble of state vectors from i=1 to Ne and the Y bar represents the average of the state vector matrix Y through each ensemble.

The most important observation when comparing the Ensemble Kalman Filter to the Kalman Filter method is that the calculation of the product of the sensitivity matrix and covariance matrix of the model variables can readily be calculated by just using the covariance of the ensemble of the state vector. This is important because this calculation does not rely on the calculation of adjoints to calculate the sensitivity matrix.

2.4 ENSEMBLE KALMAN FILTER RELATION TO ADJOINT THEORY

The adjoint method of calculating gradients with respect to an objective function has succinct advantages given that the reservoir simulator is able to solve the adjoint system of equations, but there also exist a supreme disadvantage if the reservoir simulator is unable to solve the adjoint equations. The Ensemble Kalman filter holds promise in that the calculation of gradients is done primarily by the incorporation of the covariance matrix of the state variable (through the assimilation of an ensemble of state vector realizations) in combination with the measurement matrix. The Ensemble Kalman Filter has one clear advantage over the adjoint method of gradients because the Ensemble Kalman Filter does not have to rely on solving the adjoint equations for the Lagrangian gradients. One only needs to adequately define the objective function for the control problem and apply the Ensemble Kalman Filter process to calculate the gradient to perform the optimization. Yan Chen, Dean Oliver, and Dongxiao explored the option of Ensemble Kalman filter for the water flood optimization of a five spot pattern consisting of 9 producers and 4 injectors (Chen, et al., 2008). In a similar approach Nwaozo also showed that optimization can be performed using an Ensemble Kalman Filter (Nwaozo, 2006). These Ensemble Kalman Filter approaches do show that optimization of an objective function is possible without using adjoints. This approach trumps the adjoint based optimization because it does not rely on the dynamics of the simulator.

2.5 OPTIMIZATION TECHNIQUE

The optimization technique used in this project is a variant of the Ensemble Kalman filter. It was first proposed by Nwaozo (Nwaozo, 2006). The method has been adapted for this research and the following explanation has been adjusted to describe the surfactant flood controls. This technique can be explained by first defining a standard set of controls x for a system through time interval t=1 until t=TSTEP.

$$x = \left[\{C_{P1}, C_{P2}, C_{P3}, C_{P4}, R_{INJ1}, S_{CONC}\}_{t=1}, \ldots, \{C_{P1}, C_{P2}, C_{P3}, C_{P4}, R_{INJ1}, S_{CONC}\}_{t=TSTEP} \right]^T \quad \textbf{(17)}$$

C_{P1}, C_{P2}, C_{P3}, and C_{P4} indicate the controls of the producer wells (bottomhole pressure control or total liquid control) in the 5 spot configurations; R_{INJ1} refers to the water injection rate of the injector; and S_{CONC} refers to the concentration of the surfactant in the injection well. The control setting that maximizes the objective function, $S(x)$, can be written as the following:

$$S(x) = g(x) - \frac{\alpha}{2}\left(x - x_p\right)^T C_x^{-1}\left(x - x_p\right) \quad \textbf{(18)}$$

where $g(x)$ is the output that needs to be optimized (NPV), α is the weighting factor, x is the new set of controls, x_p is the prior set of controls, and C_x is the covariance matrix of the new set of controls. The objective function can be approximated locally at $x = x'$ by using the following expression:

$$F(x' + \delta x) = S(x') + \gamma^T \delta x + \frac{1}{2}\delta x^T H \delta x \quad \textbf{(19)}$$

where $F(x'+\delta x)$ is the local quadratic approximation of $S(x)$, γ is equal to $\nabla S(x)$, H is the Hessian matrix and is equal to $\nabla[\nabla S(x)]^T$, and δx is the incremental step change in the controls. The δx that maximizes $F(x'+\delta x)$occurs at $F=0$. This observation can be written in the following equation:

$$\gamma + H\delta x = 0 \quad \textbf{(20)}$$

To numerically find δx that maximizes the objective function approximation, a Newton iteration can be used (*it* indicates iteration):

$$H_{it}\delta x^{it+1} = -\nabla S_{it} \quad \textbf{(21)}$$

This δx^{it+1} can be used to update the controls for the next iteration.

$$x^{it+1} = x^{it} + \delta x^{it+1} \tag{22}$$

The gradient for the it^{th} iteration is the following:

$$\nabla S(x) = G(x^{it}) - \alpha C_x^{-1}(x^{it} - x_p) \tag{23}$$

$G(x^{it})$ represents the matrix of derivatives of the objective function with respect to the controls and is a measure of how x affects $g(x)$. The individual components of this matrix are given by the following:

$$G_{i,j} = \frac{\partial g_i}{\partial x_j} \tag{24}$$

The Hessian matrix can then be approximated by

$$H \approx \nabla G(x^{it}) - \alpha C_x^{-1} \tag{25}$$

An assumption can be made by regarding the rate of change of x with respect to $g(x)$. This rate of change, $\nabla G(x^{it})$, can be approximated as equal to zero. This leaves the

Hessian approximation as the following:

$$H \approx -\alpha C_x^{-1} \tag{26}$$

Using the approximated Hessian matrix and the gradient for the it^{th} iteration the Newtonian iteration can be rewritten as the following:

$$-\alpha C_x^{-1}\delta x^{it+1} = -\left[G(x^{it}) - \alpha C_x^{-1}(x^{it} - x_p)\right] \tag{27}$$

Solving for δx^{it+1} leaves

$$\delta x^{it+1} = \frac{1}{\alpha} C_x G(x^{it}) - (x^{it} - x_p) \tag{28}$$

Substituting δx^{it+1} into the updated controls leaves

$$x^{it+1} = x^{it} + \frac{1}{\alpha} C_x G(x^{it}) - x^{it} + x_p \quad \textbf{(29)}$$

After cancellation, the updated controls can be expressed as

$$x^{it+1} = \frac{1}{\alpha} C_x G(x^{it}) + x_p \quad \textbf{(30)}$$

The updated controls for the upcoming iteration can be thought of as the controls in the Newtonian iteration that maximizes the objective function approximation *F(x'+δx)*. The main task in using this optimization technique is in finding how to determine $C_x G(x^{it})$. This variable can be determined by first generating *Ne* number of realizations (or ensembles) of control *x* and the NPVs associated with these controls such that a new *Y* matrix can be formed as in the following equation.

$$Y = \begin{bmatrix} \begin{Bmatrix} C_{P1\,i=1} \\ C_{P2\,i=1} \\ C_{P3\,i=1} \\ C_{P4\,i=1} \\ R_{INJ1\,i=1} \\ S_{CONC\,i=1} \end{Bmatrix}_{t=1} & \begin{Bmatrix} C_{P1\,i=2} \\ C_{P2\,i=2} \\ C_{P3\,i=2} \\ C_{P4\,i=2} \\ R_{INJ1\,i=2} \\ S_{CONC\,i=2} \end{Bmatrix}_{t=1} & \cdots\cdots & \begin{Bmatrix} C_{P1\,i=Ne} \\ C_{P2\,i=Ne} \\ C_{P3\,i=Ne} \\ C_{P4\,i=Ne} \\ R_{INJ1\,i=Ne} \\ S_{CONC\,i=Ne} \end{Bmatrix}_{t=1} \\ \begin{Bmatrix} C_{P1\,i=1} \\ C_{P2\,i=1} \\ C_{P3\,i=1} \\ C_{P4\,i=1} \\ R_{INJ1\,i=1} \\ S_{CONC\,i=1} \end{Bmatrix}_{t=2} & \begin{Bmatrix} C_{P1\,i=2} \\ C_{P2\,i=2} \\ C_{P3\,i=2} \\ C_{P4\,i=2} \\ R_{INJ1\,i=2} \\ S_{CONC\,i=2} \end{Bmatrix}_{t=2} & \cdots\cdots & \begin{Bmatrix} C_{P1\,i=Ne} \\ C_{P2\,i=Ne} \\ C_{P3\,i=Ne} \\ C_{P4\,i=Ne} \\ R_{INJ1\,i=Ne} \\ S_{CONC\,i=Ne} \end{Bmatrix}_{t=2} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \begin{Bmatrix} C_{P1\,i=1} \\ C_{P2\,i=1} \\ C_{P3\,i=1} \\ C_{P4\,i=1} \\ R_{INJ1\,i=1} \\ S_{CONC\,i=1} \end{Bmatrix}_{t=TSTEP} & \begin{Bmatrix} C_{P1\,i=2} \\ C_{P2\,i=2} \\ C_{P3\,i=2} \\ C_{P4\,i=2} \\ R_{INJ1\,i=2} \\ S_{CONC\,i=2} \end{Bmatrix}_{t=TSTEP} & \cdots\cdots & \begin{Bmatrix} C_{P1\,i=Ne} \\ C_{P2\,i=Ne} \\ C_{P3\,i=Ne} \\ C_{P4\,i=Ne} \\ R_{INJ1\,i=Ne} \\ S_{CONC\,i=Ne} \end{Bmatrix}_{t=TSTEP} \\ NPV_{i=1} & NPV_{i=2} & \cdots\cdots & NPV_{i=Ne} \end{bmatrix} \quad \textbf{(31)}$$

Or, to simplify,

$$Y = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_{Ne} \\ \text{NPV}_1 & \text{NPV}_2 & \text{NPV}_3 & \cdots & \text{NPV}_{Ne} \end{bmatrix} \quad \textbf{(32)}$$

where x represents the controls through every time interval.

Y is known as the ensemble state matrix and can be used to find the mean of the realizations through each control and NPV.

$$\bar{Y} = \frac{1}{Ne} \begin{bmatrix} \sum_{i=1}^{Ne} x_i \\ \sum_{i=1}^{Ne} \text{NPV}_i \end{bmatrix} \quad \textbf{(33)}$$

The covariance matrix of Y can then be calculated using the following expression:

$$C_y = \frac{1}{Ne-1}\left(Y-\bar{Y}\right)\left(Y-\bar{Y}\right)^T \quad \textbf{(34)}$$

The expression $C_x G(x^k)$ is related to $C_y M^T$ where M is a matrix represent by $M=[\boldsymbol{0}|\boldsymbol{I}]$. $\boldsymbol{0}$ is a N_d by N_y-N_d, matrix where N_d is the number of measurements (equal to 1 for this research since NPV is the only objective function), and N_y is the number of variables in the state vector (equal to 6TSTEP+1). $\boldsymbol{I}$ is a matrix consisting of 1s with dimensions N_d by N_d. Using the reformed state vector the updated controls for the next iteration can be found using the following expression for one realization.

$$y^{it+1} = \frac{1}{\alpha} C_y M^T + y_p \quad \textbf{(35)}$$

or

$$\begin{bmatrix} x \\ NPV \end{bmatrix}^{it+1} = \frac{1}{\alpha} C_y M^T + \begin{bmatrix} x \\ NPV \end{bmatrix}_p \quad \textbf{(36)}$$

The updated controls are the x vector in the it+1 matrix. The NPV in the it+1 matrix is not included in the controls because the NPV is not a control but a result of calculating the updated y vector.

2.6 SURFACTANT FLOOD OPTIMIZATION HISTORY

There have been various approaches and best practices to optimizing the economics of a surfactant flood. Some have treated the problem as a purely physical process disregarding the importance of the economics while others have treated it as a business decision incorporating the physics of the process.

Porzucek and Ramirez (Porzucek, et al., 1988a) were among the first to iterate that surfactant flooding is an economic problem. They argued that authors Vinatieri and Flemming (Porzucek, et al., 1988a) did not approach the optimization problem correctly because they were only concerned with optimizing the static phase behavior experiments. Porzucek and Ramirez argue that it is not purely a physical problem but an economical problem also. They continued in their analysis of past work by critiquing the work that argues for optimal salinity as vital in designing optimal surfactant floods (Porzucek, et al., 1988a). Assimilating salinity gradients in the design of surfactant floods involves forcing the reservoir to an ideal phase behavior that maximizes oil recovery with minimum surfactant retained in the residual reservoir oil. Porzucek and Ramirez argue that incorporating salinity gradients based on laboratory experiments is important in the design of surfactant floods, but state that it is unclear if optimizing the salinity has any noticeable effect on the best economic conditions for the surfactant flood. Phase behavior is a part of the surfactant flood process, and therefore should be carefully considered; but any optimization that does not incorporate the economics of the process is not a true optimization. This is because project decisions are based on not only the overall physics of the process but also the economics. This means that if a process phase behavior is optimized without consideration of the economics the likelihood of the process's economic success is unknown. This uncertainty may result in a failed process due to poor economics.

Prior to the work of Porzucek and Ramirez, surfactant flood optimization with an emphasis on the economics was not attempted until 1981. Researchers at The University of Colorado successful optimized an objective function that represents the cost of chemicals injected minus the value of the recovered oil. The model incorporated the surfactant, salt, and polymer concentrations but the only control variable was the surfactant concentration. These researchers used concepts from optimal control theory and a one dimensional surfactant/polymer flooding simulator that treated the reservoir fluids as two liquid phases (aqueous and oil). This research was significant because the optimized solution detailed the necessary surfactant concentration needed to have successful economics from the overall surfactant flooding process.

Porzucek and Ramirez extended the work of the researches at the University of Colorado by working in two dimensions using the concept of a compositional streamline simulation for the simulator. Their objective function was the overall concentration of surfactant, polymer, anions, calcium, and alcohol in the injection well as a function time while their objective function was the profit from the surfactant/polymer flood (maximizing the profit) (Porzucek, et al., 1988b). The work from Porzucek and Ramirez was significant because it merged the economics and physics behind surfactant flooding and utilized them to maximize the profit.

There has also been work done optimizing surfactant floods using approaches besides control theory. Wu, Vaskas, Delshad, Pope and Sepehrnoori used an iterative approach to the optimization of the NPV of a surfactant flood (Wu, et al., 1996). The simulator they used was UTCHEM's compositional reservoir simulator and their control variables were the slug size and concentration of surfactant. The iterative approach they used consisted of first keeping the surfactant concentration constant while varying the slug size. This was done for several surfactant concentrations until the optimum surfactant concentration was found at an optimal slug size. Using these optimal parameters, they then performed a sensitivity analysis on the economics. Part of this process can be seen in Figure 1 (Wu, et al., 1996) in which these researches determine the best surfactant concentration that gives the optimal NPV for a constant slug and constant economic environment (constant oil price and operating cost). Once the optimal surfactant concentration and slug size was chosen the researchers performed sensitivity analysis for several economic scenarios.

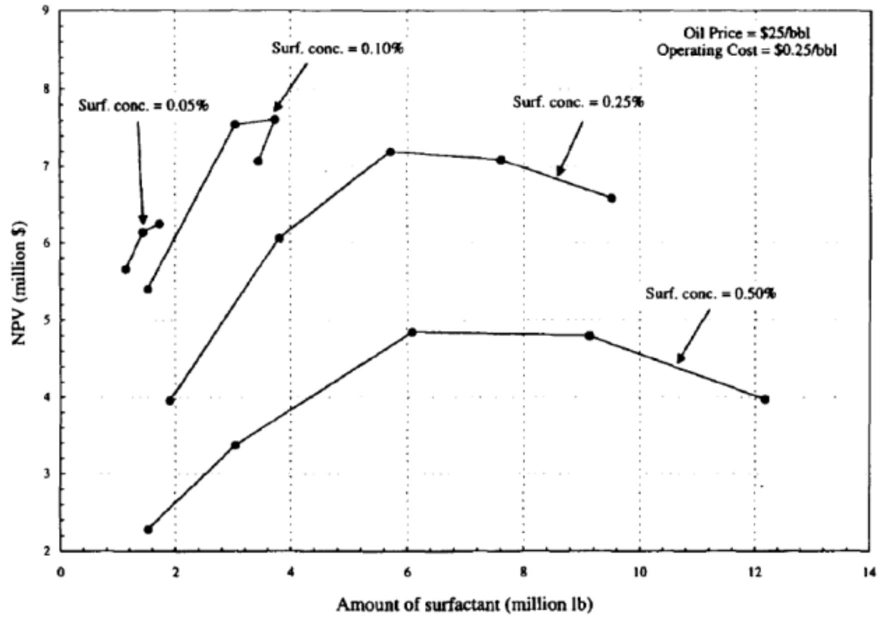


Figure 1: Iterative Approach to Optimization

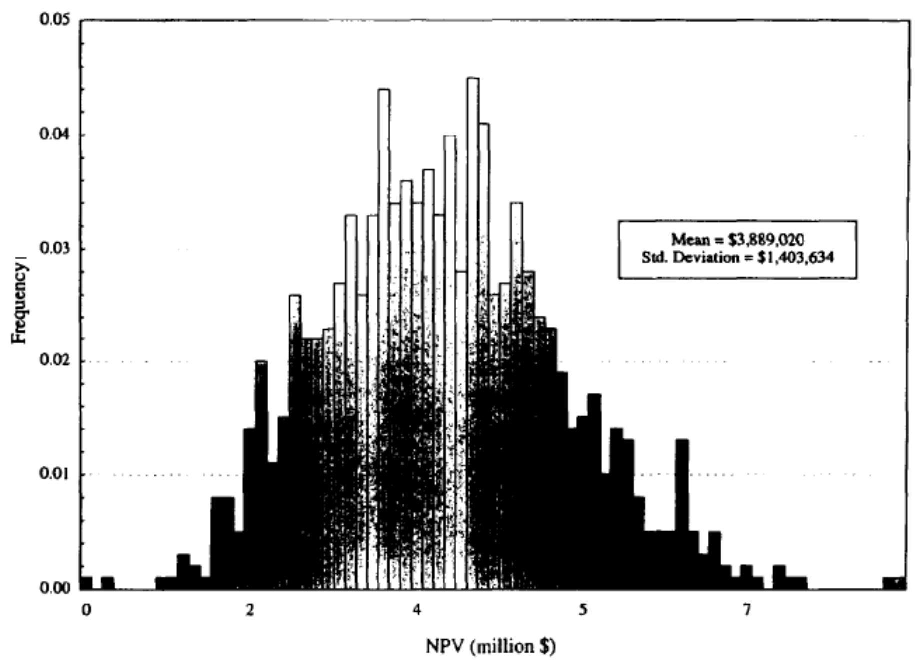


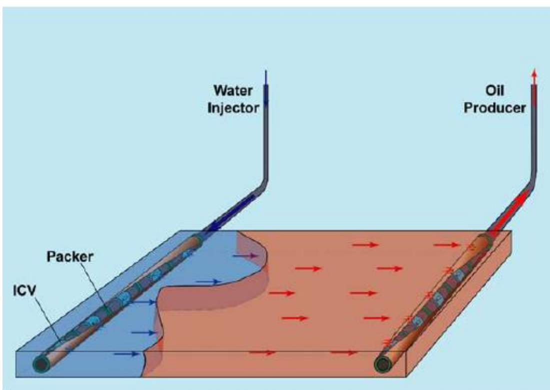
Figure 2: Monte Carlo Sampling to Determine Distribution of NPV

The sensitivity analysis was performed by sampling from probability distributions values that represent possible economic scenarios for oil. The authors then were able to generate a probability density function, like Figure 2 (Wu, et al., 1996), which gave an estimate of the possible NPV that may occur.

2.7 WATER FLOOD OPTIMIZATION HISTORY

Optimization of water floods has been an evolving topic in the oil industry with water flooding one of the most discussed topics in oil recovery optimization outside of primary depletion. Surfactant flood optimization is similar to water flood optimization in that only surfactant concentration is added to the optimization parameters for a surfactant flood. Because of this, there is a benefit of studying the history of water flood optimization so that the same principles can be applied to surfactant flood optimization. In the waterflood optimization procedures researched, the main similarity is that there is a gradient that needs to be calculated to optimize the NPV. The most popular method to calculate this gradient is the adjoint method. The important aspect of this method is that it relies on calculating the adjoints within the reservoir simulator, and thus relies on specific access to the reservoir simulator computer code. This is inconvenient because these adjoints are used to calculate the gradient for optimization. Regardless of this, it is still important to understand the pros and cons of different optimization methods such as the adjoint method for a waterflood.

Naevdal, Brouwer, and Jansen (Naevdal, et al., 2006) were successful in demonstrating the use of adjoint based methods in closed-loop optimization of a water flood. More specifically, they applied the Ensemble Kalman Filter (to history match) in conjunction with adjoint based optimization to improve the performance of a horizontal water flood. A schematic of their setup and procedure can be seen in Figure 3 (Naevdal, et al., 2006). In their work they saw significant improvement in the objective function which was the NPV.



1. Generate an initial ensemble for the Kalman filter.
2. Generate the first measurements by running the simulator over the measurement interval with the true permeability and equal production/injection rates in each well segment.
3. Run the reservoir simulator for each of the members of the ensemble Kalman filter over the measurement time interval with the same production/injection rates as used for the true permeability.
4. Update all the ensemble members by taking into account the recent measurements.
5. Compute the mean of the ensemble members and use that as the current estimate of the reservoir. Compute an optimal production/injection strategy by running the reservoir simulator and the adjoint model over the entire reservoir life several times, while iteratively updating the control strategy.
6. Apply the production/injection strategy computed above and run the simulator with the true permeability up to the next point in time where measurements are going to be assimilated, using the dynamic states from the previous run as initial conditions. Save the dynamic states at this point. Return to step 3.

(a) (b)

Figure 3: Horizontal Water Flood Setup and Procedure for History Matching and Optimization: (a) Horizontal Water Flood Setup, (b) Procedure for History Matching Using Kalman Filter and Optimization Using Adjoint Based Optimization

The only drawback to their approach is that access to the simulator code is needed to compute the adjoints that are needed to perform the optimization.

Lorentzen, Berg, Naevdal, and Vefring (Lorentzen, et al., 2006) were successful in demonstrating how the Ensemble Kalman filter principles could be used in the dynamic optimization of a vertical water flood by controlling the choke settings. The Ensemble Kalman Filter works by forcing a system to match a series of observations. Lorentzen, Berg, Naevdal, and Vefring utilized this concept by defining an upper limit of the objective function which was the cumulative oil or NPV.

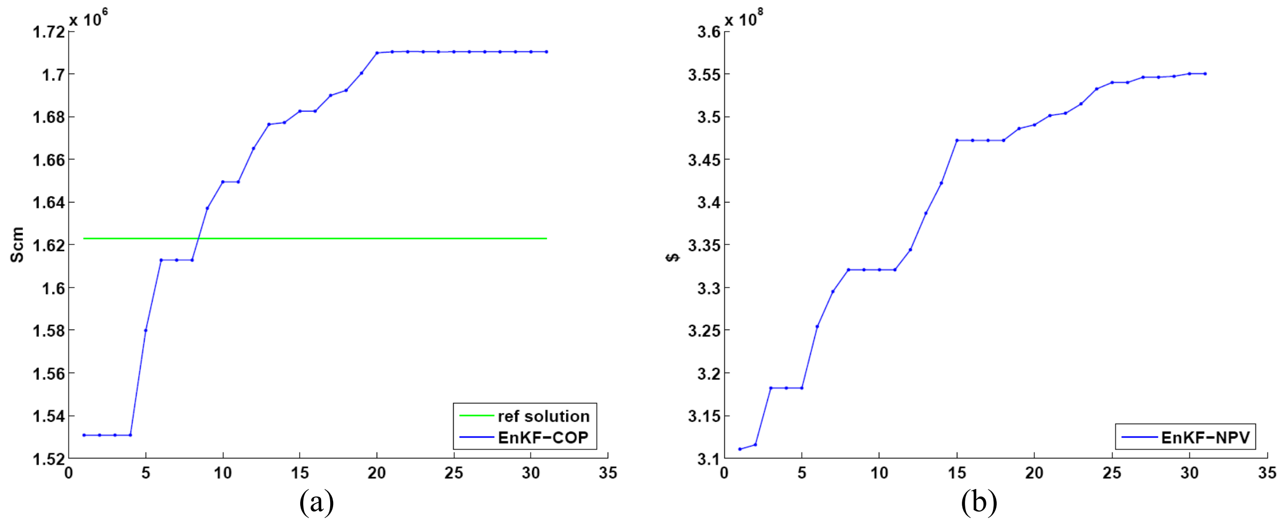


Figure 4: Ensemble Kalman Filter Optimization: (a) Cumulative Oil Production Versus Iterations, (b) Net Present Value Versus Iterations

The authors saw significant improvement in the NPV and cumulative oil when they utilized the modified Ensemble Kalman Filter (see Figure 4); the only drawback to this approach is that the method is dependent on defining an upper limit on the NPV. Ideally optimization algorithms should not have to rely on setting an upper limit on the NPV because the whole goal of maximizing the NPV is to search for the highest NPV possible; defining an upper limit on the NPV is inconvenient if the user doesn't know a possible maximum.

More recently, Nwaozo (Nwaozo, 2006) carried Lorentzen, Berg, Naevdal, and Vefring's approach further by avoiding the vague task of defining an upper limit of the NPV. Nwaozo used the Ensemble Kalman Filter as a means to approximating the gradient needed to update the controls for the next iteration for the optimization of the NPV. Nwaozo applied this optimization scheme to a 5 spot vertical well water flood (with bottomhole pressure for the four producers for every time interval as the control) and although there was significant improvement of the NPV more work had to be done in defining the initial ensemble of controls. A prior is defined as the starting point for the optimization and an initial ensemble are realizations of the prior. These realizations of the prior in Ensemble Kalman Filter terminology generally are based on physical evidence. For example in the history matching problem of matching the permeability field to observations such as the watercut, the initial ensemble of the permeability is created from a distribution of the permeability of the wells in the field. In using the Ensemble Kalman Filter for optimization, each realization created has to be based on previous physical evidence. Nwaozo did not create his ensemble based on physical evidence and assumed that the initial ensemble of bottomhole pressure controls could be calculated using a Gaussian covariance function which is essentially random (see Figure 5) and has no resemblance in relation to classical petroleum engineering production theory that describes bottom pressure as an inflow performance relationship.

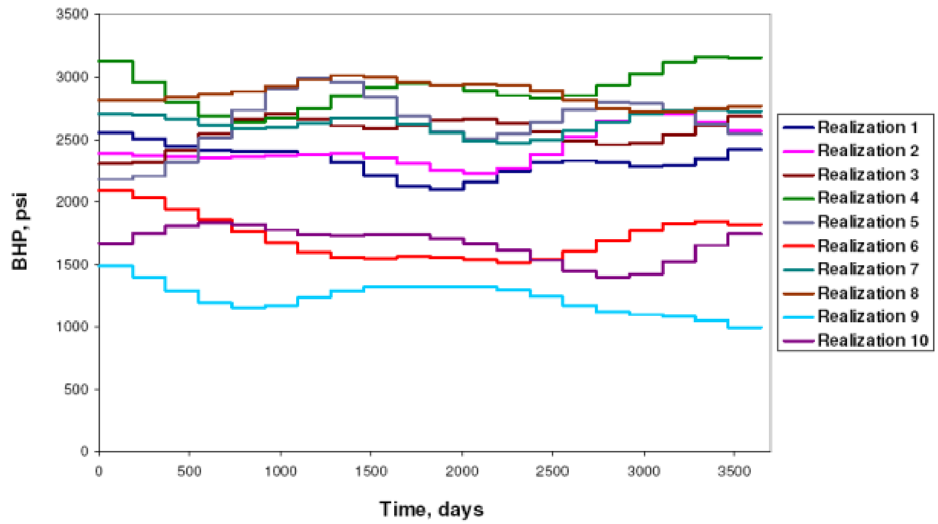


Figure 5: Prior BHP Profile for Producer 1 Defined by Gaussian Covariance Function

Another problem with Nwaozo's work is in the selection of the weighting factor used to advance the controls for the next optimization iteration. Nwaozo chose an initial large weighting factor and reduced the weighting factor at a constant step size until the NPV was maximized for an optimization iteration. The problem with this approach is that it relies on knowledge on the approximate location of the ideal weighting factor, because if one starts reducing the weighting factor far away from the optimal it may take inconveniently long times to find the optimal weighting factor. Nwaozo had a good idea on what initial weighting factor and step size to choose but did not divulge how to obtain these parameters. It is theorized that Nwaozo, by trial and error, tried several initial weighting factors and step sizes until he found the parameters that determined the optimal weighting factor that gave the highest mean NPV. An example showing the process of selecting the optimal weighting factor or "Alpha" can be seen in Figure 6.

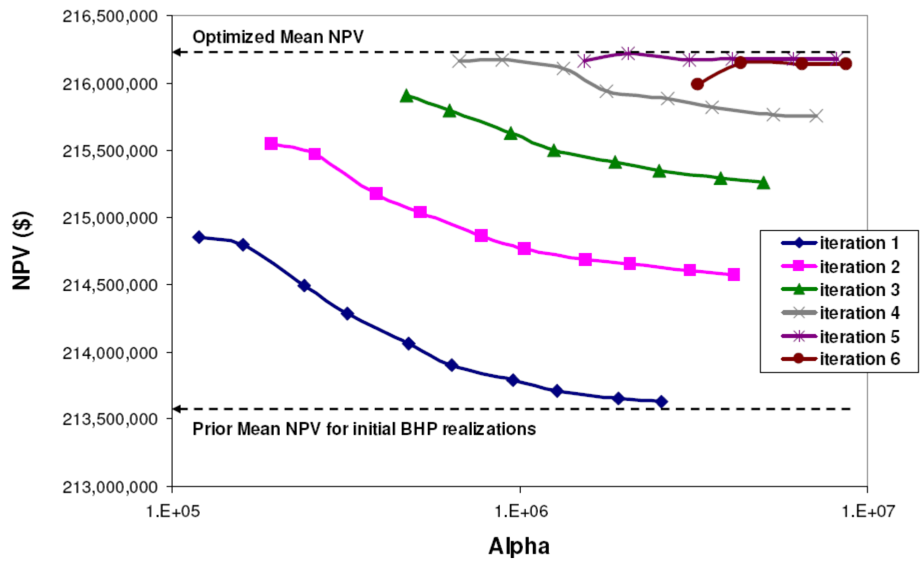


Figure 6: Selection of Weighting Factor Using Nwaozo's Approach

Although there are some problems with Nwaozo's approach to the optimization of a water flood, the method is adequate for optimization if the prior has a physical meaning and that the weighting factor can be found in adequate time. Nwaozo's approach is the one utilized in this research, but more control variables were used and they directly relate to the surfactant flood process by using concepts such as inflow performance relationship and decline curve analysis. Also, a new method was developed that constrained the controls to user defined bounds during the optimization. In addition to this, a different approach to finding the weighting factor was utilized that did not really on guessing the approximate location of the weighting factor that optimizes the NPV. The new approach specifies a large interval of possible weighting factors and rapidly converges on the global optimal weighting factor.

The purpose of this research is to optimize the performance of a surfactant flood and then to incorporate probabilistic realizations of permeability fields and probabilistic economics. The concepts learned from previous surfactant flood and waterflood projects have been used in this research.

3. RESERVOIR MODEL DESCRIPTION

The reservoir model created for this work was developed to model a two dimensional 320 acre heterogeneous field. This section addresses the work done to generate the heterogeneous fields and the reservoir properties in the simulation file. The reservoir created is purely synthetic and was created for the sake of studying optimization. This section is divided into two subsections. The first subsection describes how to create the porosity fields using the Stanford Geostatistical Modeling Software (SGEMS) (Remy, 2004). The second subsection describes the reservoir parameters used in this work.

3.1 SEQUENTIAL GAUSSIAN SIMULATION ANALYSIS USING SIMPLE KRIGING

Sequential Gaussian Simulation using Simple Kriging was performed on a 320 acre porosity field with 50 by 50 by 1 dimensions. If there are any questions regarding the theoretical frame work of Sequential Gaussian Simulation the reader is encouraged to read the SGEMS manual (Remy, 2004). The theoretical frame work of Sequential Gaussian Simulation was not the focus of this research but the application of these theories using the SGEMS software were essential in generating the porosity fields used in this research. To start the process of creating the porosity fields, it is pertinent to have raw data that shows the variation of porosity in the field. This raw data was obtained from Dr. Jafarpour’s Texas A&M University fall 2009 graduate course titled Reservoir

Characterization and Forecasting. The raw data for the porosity field can be seen in Figure 7.

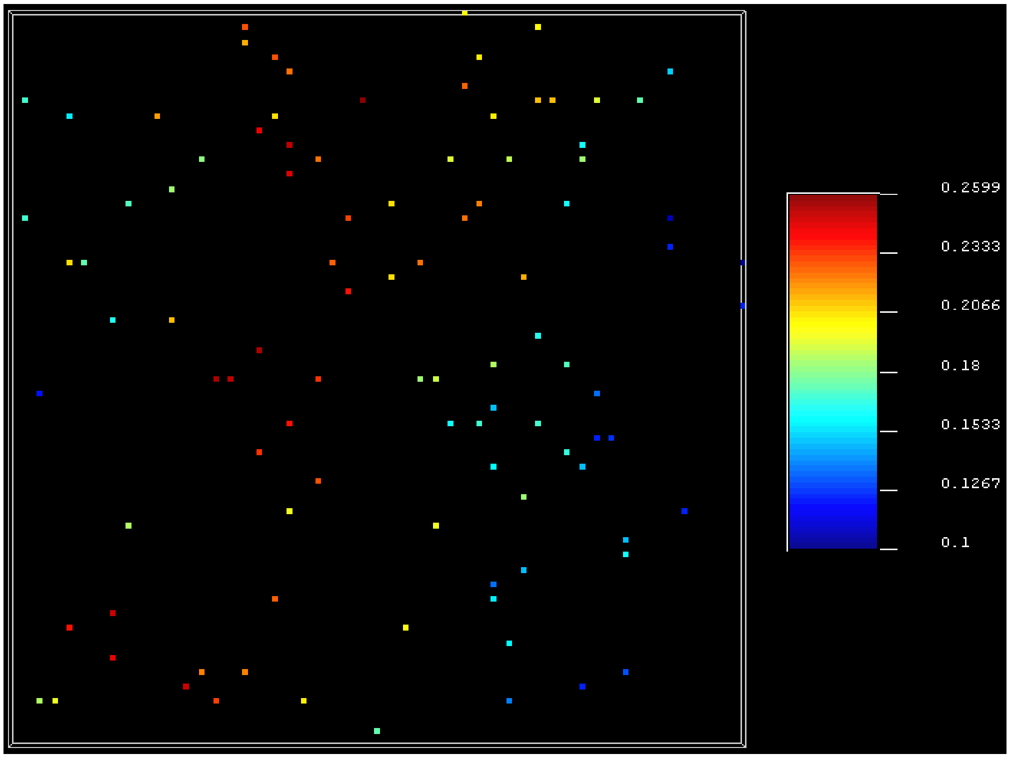


Figure 7: Raw Data of 320 Acre Porosity Field (Pore Volume Fraction)

Observing the data it can be seen that there is a wide distribution of porosity in the field with location correlation. There is an obvious trend in the North to South direction and East to West direction. To account for the wide distribution of data, a histogram and cumulative distribution function was created in SGEMS to train probabilistically the realizations of porosity fields to conform to the distribution of the data. The histogram that was created from the raw data of porosity can be seen in Figure 8.

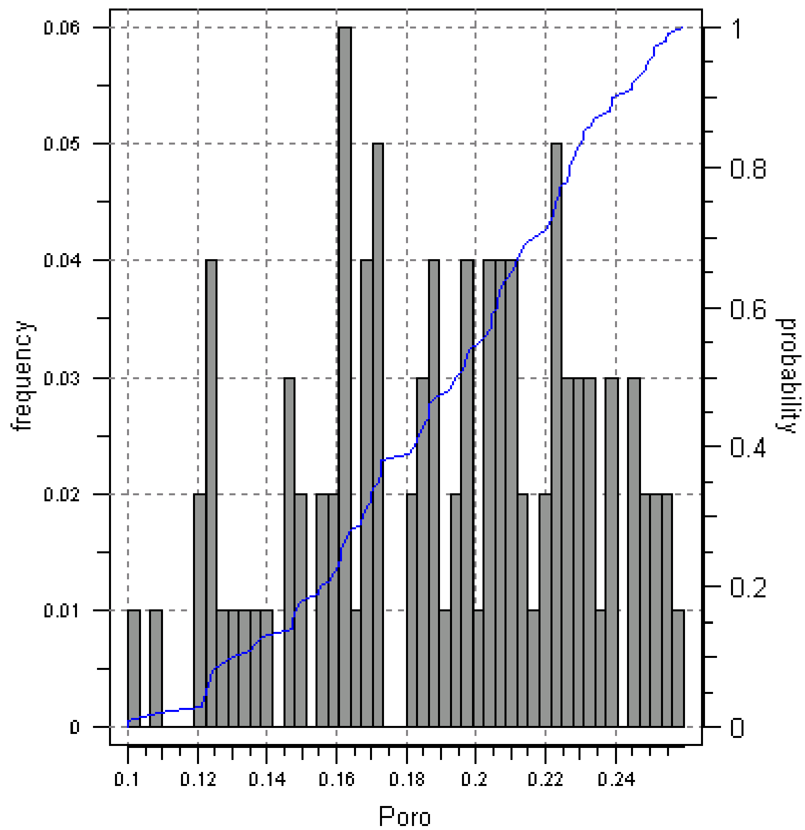


Figure 8: Histogram and Cumulative Distribution Function of Raw Data of Porosity Field

The raw data was then transformed into the Gaussian domain by performing a normal score transform using SGEMS. The purpose of doing this is to remove the influence of the mean when generating the realizations of the porosity fields. The normal score transformed raw data can be seen in Figure 9.

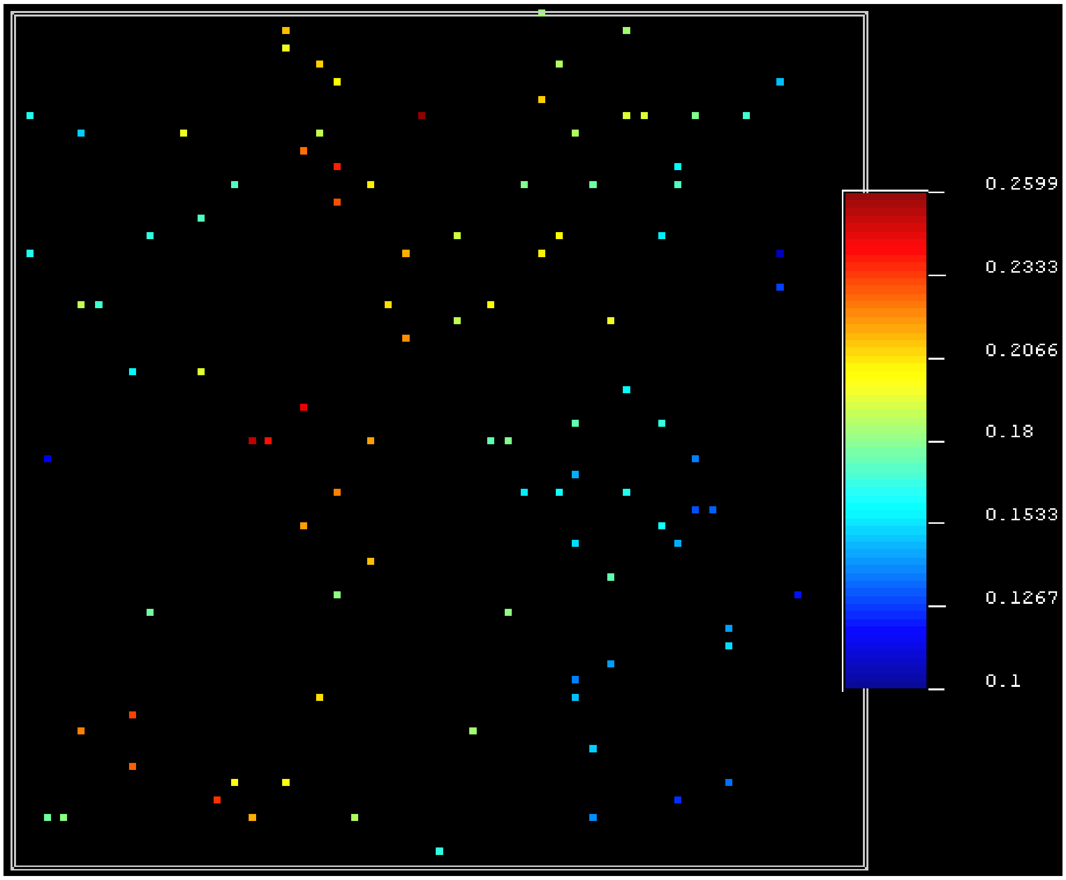
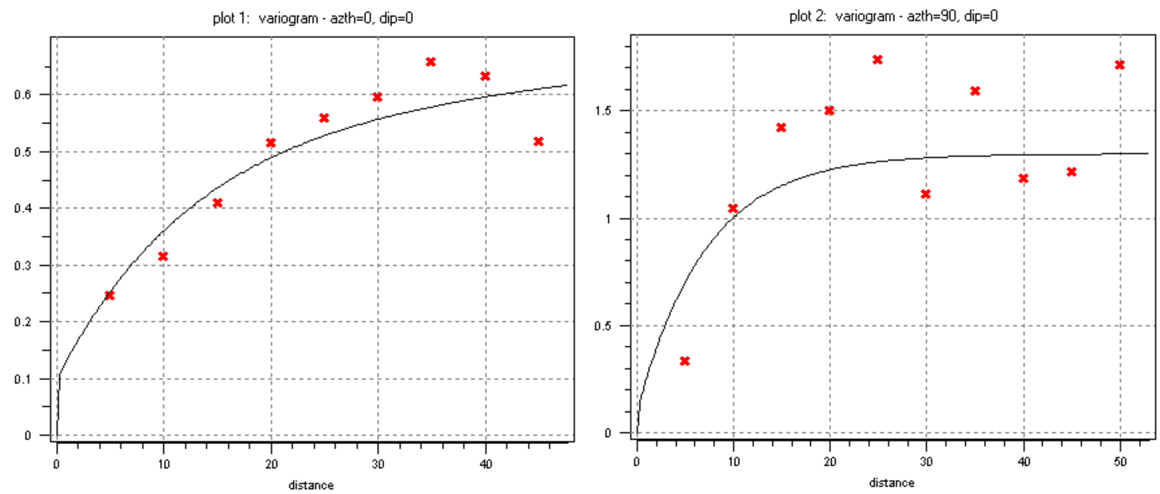


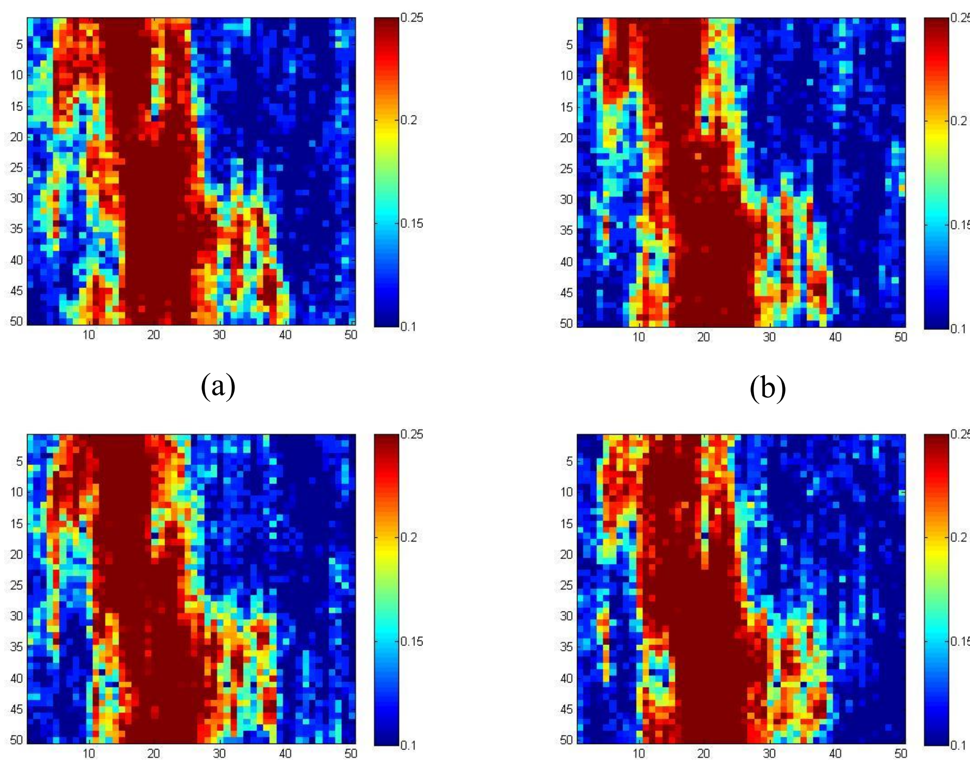
Figure 9: Normal Score Transformed Data of Porosity Field (Pore Volume Fraction)

To account for the North South and East West trends in the field a variogram was created in SGEMS based on the normal score transform of the raw data. A variogram is a measure of dissimilarity between two random variables, and is used to model spatial variability (Jafarpour, 2008f). The variogram fit that models the North South and East West directions that was created based on the normal score transformed data can be seen in Figure 10. Sequential Gaussian Simulation using Simple Kriging was then performed to create 1000 realizations of porosity fields. Four realizations of porosity fields can be seen in Figure 11.



(a) (b)

Figure 10: Variogram: (a) East West Direction, (b) North South Direction



(a) (b)

(c) (d)

Figure 11: Example Porosity Realizations: (a) Porosity Realization 1, (b) Porosity Realization 2, (c) Porosity Realization 3, (d) Porosity Realization 4

The color map on the right of each figure represents the porosity. The vertical and horizontal axes represent the grid blocks (each grid block has dimensions of 74.6705ft by 74.6705 ft and the total field is 320 acres). From observing the realizations of porosity fields, it can seen that all of the realizations honor the trend of the raw data porosity field while each having a unique representation. The 1000 realizations were used to create an arithmetic mean porosity field used for the optimization studies. The mean porosity field can be seen in Figure 12.

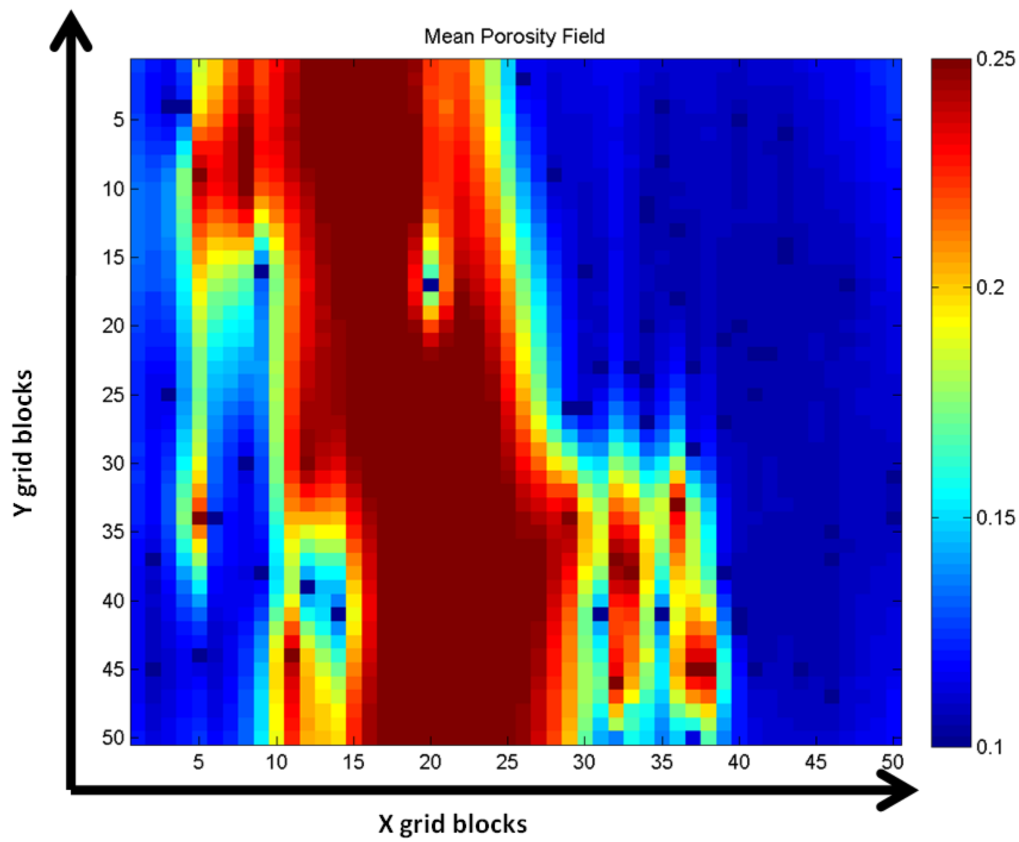


Figure 12: Mean Porosity Field (Pore Volume Fraction)

3.2 GENERAL RESERVOIR DESCRIPTION

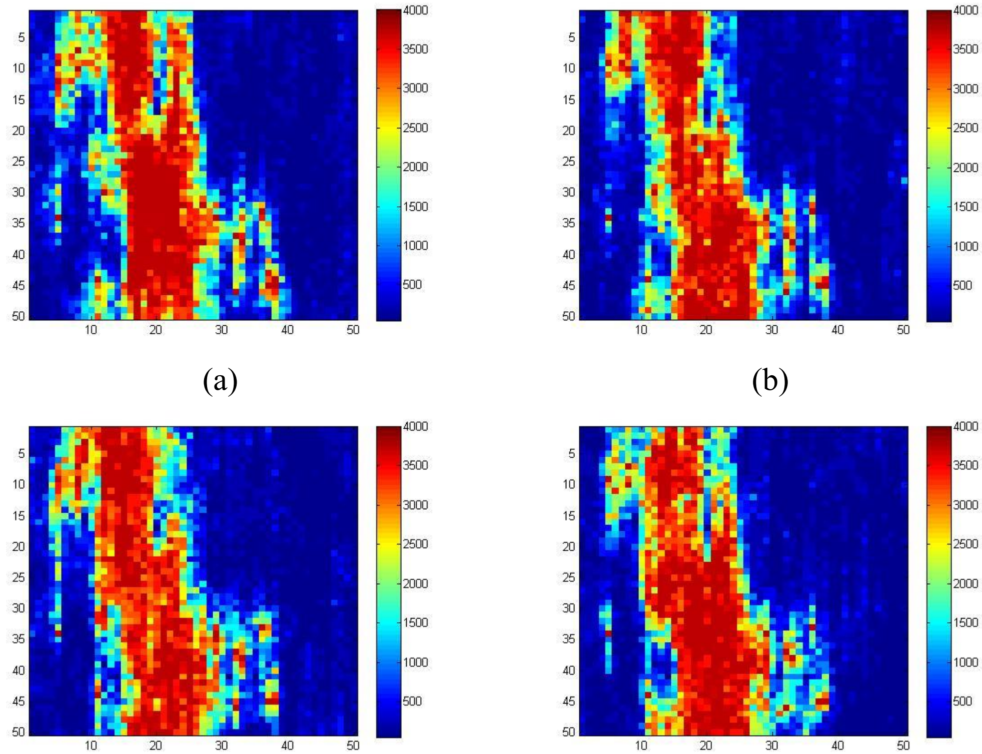
A general procedure was used to model a typical black oil field operation in Eclipse. The reservoir created was modeled for the purpose of having a system to perform an optimization upon.

3.2.1 PERMEABILITY

Multiple realizations of permeability were required for this research but hard data of a permeability field were not available. Therefore, the permeability field was created based on the porosity field realizations and the Coates and Denoo relationship (Babadagli, et al., 2002). The Coates and Denoo relationship, which is a function of irreducible water saturation, is described by the following equation:

$$k = \left[\frac{100 \cdot \phi^2 \cdot (1 - S_{wir})}{S_{wir}} \right]^2 \qquad \textbf{(37)}$$

where S_{wir} corresponds to the irreducible water saturation used for this research. Using this relationship the permeability fields that were created can be seen in Figure 13 for an irreducible water saturation of 10%.



(a) (b)

(c) (d)

Figure 13: Example Permeability Realizations in mD: (a) Permeability Realization 1, (b) Permeability Realization 2, (c) Permeability Realization 3, (d) Permeability Realization 4

The irreducible water saturation was chosen for the sake of creating permeability fields. 1000 realizations of permeability fields were created based on the 1000 realizations of porosity fields. The 1000 realizations were used to create a geometric mean permeability field used for the optimization studies. The mean permeability field can be seen in the Figure 14.

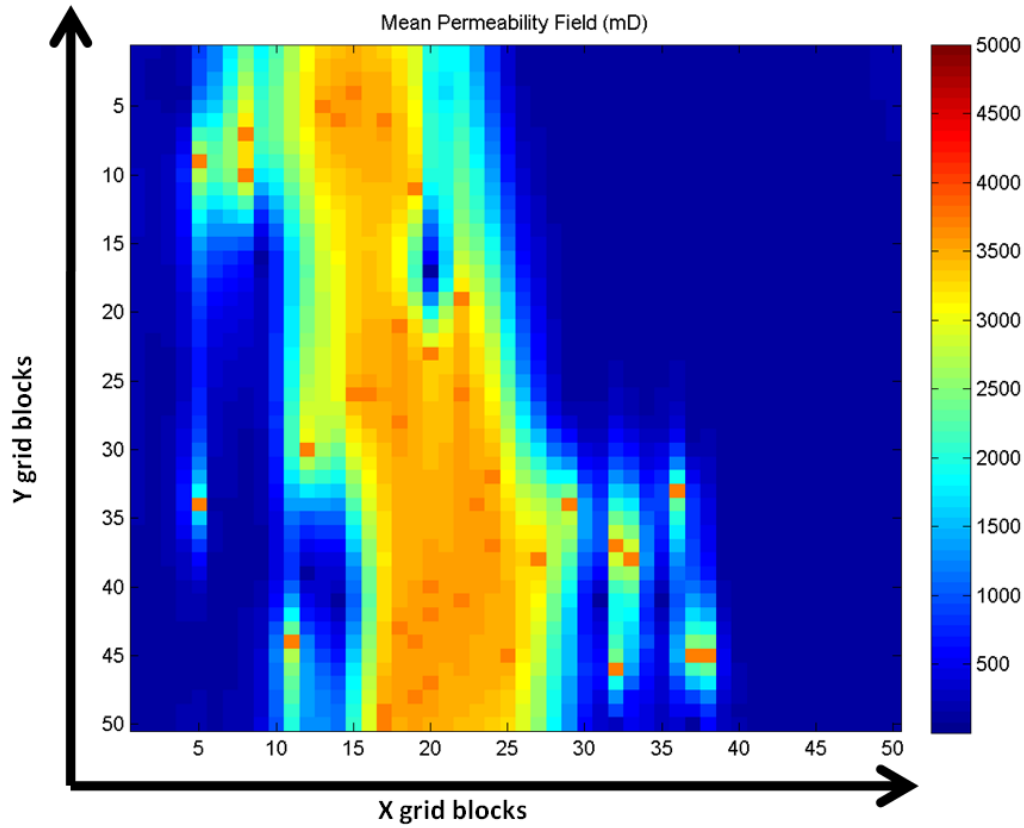


Figure 14: Mean Permeability Field

3.2.2 RELATIVE PERMEABILITY

The relative permeability for this research was based on the modified Brooks Corey model (Behrenbruch, et al., 2006) which can be seen in the following equations:

$$k_{ro} = k_{ro}^{*}\left[\frac{1 - S_w - S_{or}}{1 - S_{wir} - S_{or}}\right]^{n_o} \quad \textbf{(38)}$$

$$k_{rw} = k_{rw}^{*}\left[\frac{S_w - S_{wir}}{1 - S_{wir} - S_{or}}\right]^{n_w} \quad \textbf{(39)}$$

where the k_{ro} refers to the relative permeability of the oil, k_{rw} refers to the relative permeability to the water, S_w is the water saturation, S_{or} is residual oil saturation, n_o is the Corey exponent to the oil, n_w is the Corey exponent to the water, k^*_{rw} is the endpoint relative permeability to the water, k^*_{ro} is the endpoint relative permeability to the oil. The Brooks Corey variables that were used are listed in Table 1:

Table 1: Brook Corey Parameters

S_{wir}	0.1
S_{or}	0.3
k_{ro}*	1
k_{rw}*	1
n_o	6
n_w	4

These values were chosen to model a slightly oil wet rock (Behrenbruch, et al., 2006). The relative curves based on modified Brooks Corey model can be in Figure 15.

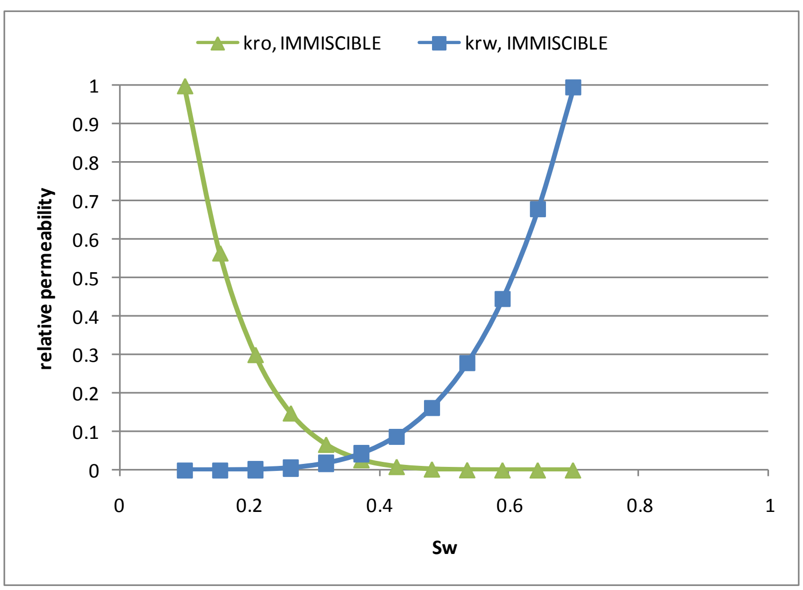


Figure 15: Relative Permeability

3.2.3 RESERVOIR PROPERTIES

The reservoir black oil simulator used in this research was the Schlumberger's Eclipse 100. For relevant details concerning the simulator used in this work, please refer to the manual (Schlumberger, 2008b). Table 2 summarizes the reservoir simulation variables that remained constant throughout this study.

Table 2: Reservoir Simulation Properties

Net pay	70 ft
Reservoir Pressure	4500 psia
Area	320 Acre
Top Depth	10000 ft
Grid Dimensions	50 X 50 X 1
Grid Block Size	DX=74.6705 ft, DY=74.6705 ft, DZ=70 ft
Formation Compressibility	5E-6 psi^{-1}
Water Compressibility	3E-6 psi^{-1}
Water Viscosity	1 cp
Water Density	62.4 lbm/ft^3
Water Formation Volume Factor	1 RB/STB
Oil Viscosity	3 cp
Oil Density	56 lbm/ft^3

The oil compressibility is not a necessary input into Eclipse 100 reservoir simulator because it is a black oil simulator and thus relies on the oil formation volume factor to calculate oil compressibility, but it is necessary for determining the initial controls for the optimization. Eclipse uses the oil formation volume factor and its reference pressure to calculate the oil compressibility (Schlumberger, 2008a), from the following equation (Dake, 2007):

$$c_o = -\frac{1}{B_o}\frac{dB_o}{dP} \tag{40}$$

where c_o is the oil compressibility, B_o is the oil formation volume factor, and P is the reservoir pressure. To use this formula it was broken down into the following finite difference form:

$$c_o = -\frac{1}{B_o^{avg}}\frac{B_o^{step+1} - B_o^{step}}{P^{step+1} - P^{step}} \tag{41}$$

where the step refers to the current value in the table, B_o^{avg} stands for the average of B_o^{step+1} and B_o^{step}. The finite difference form of the compressibility calculation was applied to the oil formation volume data used in Eclipse. The pressure and oil formation volume factor at that pressure can be seen in Table 3 along with the oil compressibility calculated at that pressure.

Table 3: Oil Compressibility Calculation

step	P, psia	B_o, RB/STB	B_o^{avg}, RB/STB	c_o, psi^{-1}
1	50	1.00001	-	-
2	4500	1	1.000005	2.2472E-09
3	10000	0.99998	7250	2.63636E-05

The total compressibility can then be calculated using the following equation (Dake, 2007):

$$c_t = c_o S_o + c_w S_w + c_f \tag{42}$$

where the c_o refers to the oil compressibility, c_w refers to the water compressibility, c_f is the formation compressibility, S_o is the oil saturation, and S_w is the water saturation.

3.2.4 WELL CONFIGURATION

All the wells in the 5 spot are vertical and completed across the 70 ft interval, with the middle of the zone at 10035 ft and well radius of .8 ft. The well locations relative to the grid used in the simulation can be seen in Table 4.

Table 4: Well Configuration

	Grid location (X,Y,Z)
P1, 1st producer location	(1,1,1)
P2, 2nd producer location	(1,50,1)
P3, 3rd producer location	(50,50,1)
P4, 4th producer location	(50,1,1)
INJ1, injector location	(25,25,1)

An aerial view of the reservoir, with the 5 spot configuration, can be seen in Figure 16.

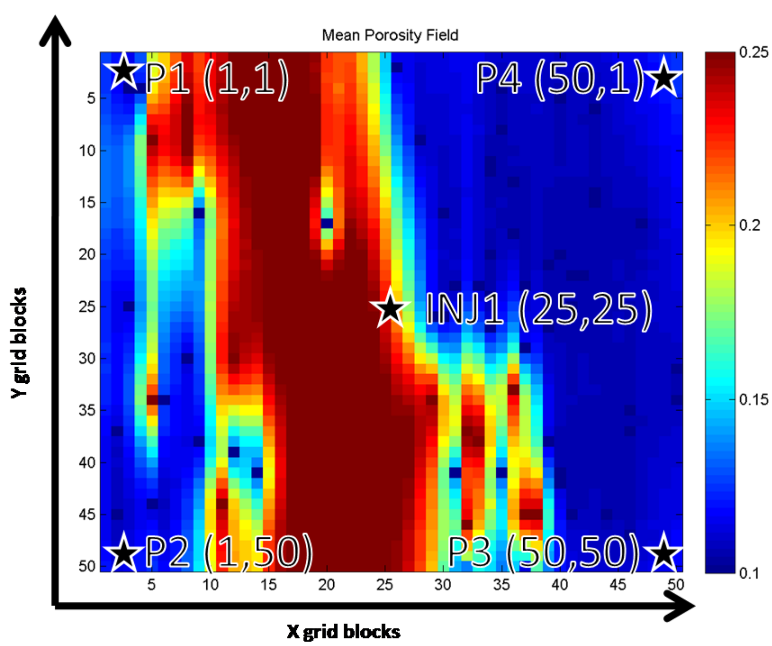


Figure 16: Well Coordinates for the 5 Spot Pattern

4. SURFACTANT MODEL

Schlumberger's Eclipse 100 Black Oil simulation software (Eclipse) was used in conjunction with the surfactant option. There are many popular simulators that can model surfactant floods using detailed chemistry. These simulators include the Computer Modeling Groups (CMG) STARS and the University of Texas Chemical Compositional Simulator (UTCHEM). Schlumberger's Eclipse 100 surfactant model differs from these simulators because it does not model the detailed chemistry, but instead models the important features of the surfactant flood in a full field basis such as the loss of surfactant due to adsorption and the dynamic immiscible and miscible relative permeability relationship (Schlumberger, 2008b). It does this by using the capillary number to adjust between a miscible state and immiscible state for each of the grid blocks (Schlumberger, 2008b), a user defined relationship between surfactant concentration and interfacial tension (Jakobsen, et al., 1994), user defined miscible relative permeability curve, and user defined adsorption isotherm. Eclipse's approach to surfactant flooding is an approximation of real micro emulsion phase and property behavior (Jakobsen, et al., 1994). Eclipse models reservoir dynamics in three phases which are oil, water, and gas. The surfactant option assumes that the surfactant exist only in the water phases and thus modifies the water phases in the black oil simulation.

For this research, laboratory data of a surfactant flood were not available, so the surfactant used was the example provided in Eclipse's Technical Manual version 2008.1 (Schlumberger, 2008b). These data were used in combination with an assumption of the immiscible and miscible relative permeability relationship. Miscibility develops dynamically when using the surfactant option and it is modeled using the surfactant option's relative permeability model. The relative permeability model is an evolution from the immiscible relative permeability curve at low capillary number to miscible relative permeability curve at high capillary number. The user describes this change as a function of the capillary number. The miscible relative permeability relationship was based on observations noted in literature (Cinar, et al., 2004). Cinar, Marquez, and Orr established a relationship between the oil wetting phase and the corresponding oil relative permeability in the three phase system of oil, gas, and water. From this relationship seen in Figure 17, it can be seen that for a low interfacial tension system the relative permeability to the oil phase is larger compared to a high interfacial tension system.

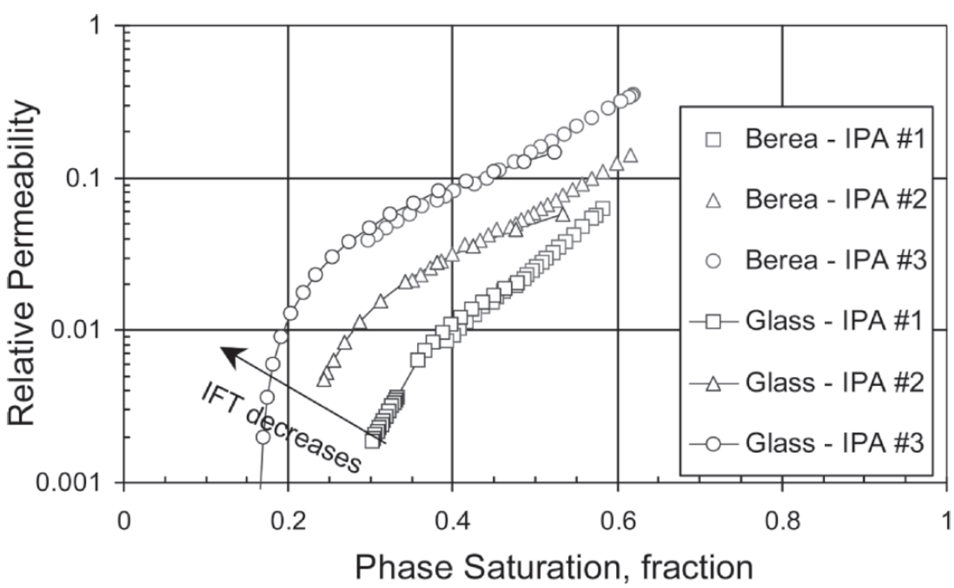


Figure 17: IFT Effect on Relative Permeability in Oil Wet System

Using this logic and the assumption that an ultra low interfacial tension will reduce the residual oil saturation, the miscible relative permeability curve was constructed for the surfactant used in this research. This curve can be seen in Figure 18.

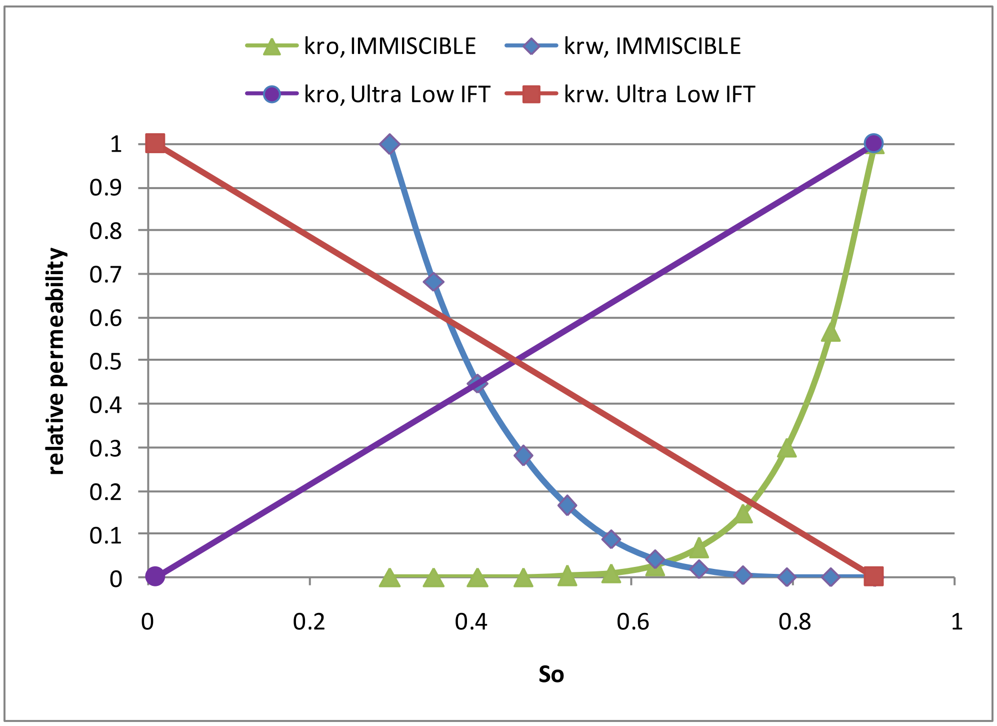


Figure 18: Relative Permeability of Immiscible and Ultra Low Interfacial Tension Conditions

The horizontal axis represents the oil saturation and the vertical axis represents the relative permeability. From this curve it can be seen that the surfactant has the possibility of lowering the residual oil saturation from 30% to 1%. The immiscible curve is just the standard relative permeability (see Reservoir Description section on relative permeability), while the ultra low interfacial tension relative permeability to the water was designed based on the assumption that the surfactant would not reduce the irreducible water saturation.

The capillary number is a dimensionless group that is the ratio of viscous to capillary forces. The standard equation to calculate capillary number is the following equation (Zhou, et al., 1993):

$$N_c = \frac{\mu \cdot q}{\sigma} \tag{43}$$

where σ stands for the interfacial tension between the oil and water phase, μ stands for the viscosity of the fluid, and the q is the Darcy flow rate. The capillary number is an indication of miscibility conditions in Eclipse (Schlumberger, 2008b). Eclipse calculates the capillary number using the following equation (Schlumberger, 2008b).

$$N_c = \frac{|k \cdot \nabla\Omega|}{\sigma} \tag{44}$$

The k times the vector differential operating on Ω product stands for the permeability times the vector differential of the flow potential and is a function of the local pressure drop in all directions (Schlumberger, 2008b). The user inputs the log of the capillary number and the corresponding state of the phase (miscible or immiscible). The user can deduce the relationship between miscible and immiscible using a capillary desaturation curve (CDC) which gives a relationship between the residual oil saturation and the capillary number (Zhou, et al., 1993). An example CDC curve can be seen in Figure 19.

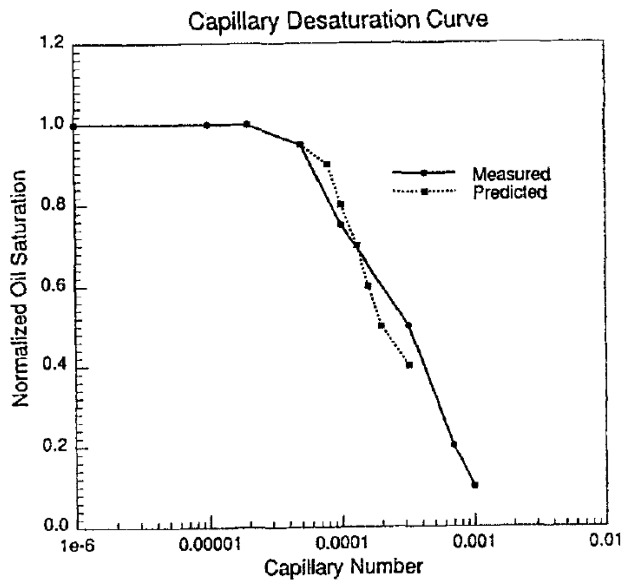


Figure 19: Capillary Desaturation Curve Example

A substantial high capillary number in conjunction with low residual oil saturation corresponds to a miscible state while a low capillary number in conjunction with high residual oil saturation corresponds to immiscible state (Schlumberger, 2008b). For this research the residual oil saturation was 30% which corresponds to an immiscible state while an ultra low interfacial tension corresponds to a residual oil saturation of 1%. Table 5 describes the state of the fluid as a function of the log capillary number.

Table 5: Capillary Number

LOG Capillary Number	Capillary Number	State
-9	0.000000001	IMMISCIBLE
-4.5	3.16228E-05	IMMISCIBLE
-2	0.01	MISCIBLE
10	10000000000	MISCIBLE

Surfactants modify the viscosity of the aqueous phase and the interfacial tension between the oleic and aqueous phases. The surfactant option requires the user to input the viscosity and interfacial tension as a function of the surfactant concentration (Schlumberger, 2008b). The viscosity and interfacial tension relationship used in this research can be seen in the Table 6 and Table 7 respectively.

Table 6: Surfactant Viscosity

Surfactant Concentration, lb/STB	Viscosity, cp
0	1
10.516	5

Table 7: Surface Tension of Surfactant

Surfactant Concentration, lb/STB	Surface Tension, lb/in
0	0.0002855
0.35053	5.71E-09
10.516	5.71E-09

Eclipse models the adsorption of surfactant as a function of the adsorption isotherm which is a function of surfactant concentration (Schlumberger, 2008b). The following equation is used to model the mass of surfactant adsorbed onto the rock (Schlumberger, 2008b).

$$M_{abs} = PV \cdot \frac{1-\phi}{\phi} \cdot \rho_R \cdot CA(S_{CONC}) \tag{45}$$

The PV stands for the pore volume in the grid block, $CA(S_{CONC})$ is the adsorption isotherm as a function of the local surfactant concentration, and ρ_R is the mass density of the rock which was 1000 lb per reservoir barrel (approximately 2.8 g/cm^3). Table 8 lists the adsorption isotherm used in this research.

Table 8: Surfactant Adsorption

Surfactant Concentration, lb/STB	Adsorption, lb surfactant/lb reservoir rock
0	0
0.35053	0.0005
10.516	0.0005

The surfactant properties used in this work were chosen for the purpose of studying a simple surfactant. To study more complex surfactants it is recommended to gather data pertaining to the surfactant properties illustrated in this section. For the purpose of studying surfactant optimization in a simulation setting, the surfactant properties chosen for this research are suitable.

5. TRANSFORMATION OF ENSEMBLE OF CONTROL MATRIX

The purpose of this section is to illustrate to the reader the newly created method developed in this research to limit the controls during the optimization process. This new method is unique and has not been developed before for optimization purposes. It is necessary to go through the derivation and to explain how to use the new proposed method of constraining the controls.

The general optimization procedure proposed by Nwaozo and used in this research is a valid way for finding the optimal solution for a variety of problems. The solutions to these problems may even have values that are negative but are valid for the systems controls. Valid controls for this research however are not negative and therefore have to be numerically bounded to ensure realistic answers. To do this the ensemble of controls was transformed to ensure user defined control constraints. This transformation was done using a normal score transformation of the Y ensemble matrix.

The transformation method used in this research is the normal score transformation (NST) method. This method consists of standardizing each control to the standard normal domain using the mean and standard deviation of the control. To standardize the control, the following relation was used for each controller element (Montgomery, et al., 2007):

$$s(c) = \frac{c - \mu_c}{\sigma_c} \tag{46}$$

where $s(c)$ corresponds to the standardization of control c (the producer control, injection rate control, or surfactant concentration control), μ_c corresponds to the mean of the control, and σ_c corresponds to the standard deviation of the control. The standardized control was used to standardize each of the six controls through each time interval therefore each controller (4 producers, one injector, and surfactant concentration) had a mean and standard deviation associated with it. The mean and standard deviation was calculated using the maximum and minimum constraint for the control. This process can be illustrated by first assuming that the maximum and minimum controls are related to the mean and standard deviation by the following system of equations:

$$c_{\min} = \mu_c - \sigma_c \tag{47}$$

$$c_{\max} = \mu_c + \sigma_c \tag{48}$$

where c_{min} refers to the minimum value of the control and c_{max} refers to the maximum value of the control. Solving the system of equations leaves the following relations for the mean and standard deviation.

$$\mu_c = \frac{c_{\max} + c_{\min}}{2} \tag{49}$$

$$\sigma_c = \frac{c_{\max} - c_{\min}}{2} \tag{50}$$

Using the standardized control the ensemble matrix Y can be redefined as the Normal Score Transform (NST) ensemble matrix Y_s^*.

$$Y_S^* = \begin{bmatrix} \left\{\begin{matrix} s(C_{P1\,i=1}) \\ s(C_{P2\,i=1}) \\ s(C_{P3\,i=1}) \\ s(C_{P4\,i=1}) \\ s(R_{INJ1\,i=1}) \\ s(S_{CONC\,i=1}) \end{matrix}\right\}_{t=1} & \left\{\begin{matrix} s(C_{P1\,i=2}) \\ s(C_{P2\,i=2}) \\ s(C_{P3\,i=2}) \\ s(C_{P4\,i=2}) \\ s(R_{INJ1\,i=2}) \\ s(S_{CONC\,i=2}) \end{matrix}\right\}_{t=1} & \cdots\cdots\cdots & \left\{\begin{matrix} s(C_{P1\,i=Ne}) \\ s(C_{P2\,i=Ne}) \\ s(C_{P3\,i=Ne}) \\ s(C_{P4\,i=Ne}) \\ s(R_{INJ1\,i=Ne}) \\ s(S_{CONC\,i=Ne}) \end{matrix}\right\}_{t=1} \\ \left\{\begin{matrix} s(C_{P1\,i=1}) \\ s(C_{P2\,i=1}) \\ s(C_{P3\,i=1}) \\ s(C_{P4\,i=1}) \\ s(R_{INJ1\,i=1}) \\ s(S_{CONC\,i=1}) \end{matrix}\right\}_{t=2} & \left\{\begin{matrix} s(C_{P1\,i=2}) \\ s(C_{P2\,i=2}) \\ s(C_{P3\,i=2}) \\ s(C_{P4\,i=2}) \\ s(R_{INJ1\,i=2}) \\ s(S_{CONC\,i=2}) \end{matrix}\right\}_{t=2} & \cdots\cdots\cdots & \left\{\begin{matrix} s(C_{P1\,i=Ne}) \\ s(C_{P2\,i=Ne}) \\ s(C_{P3\,i=Ne}) \\ s(C_{P4\,i=Ne}) \\ s(R_{INJ1\,i=Ne}) \\ s(S_{CONC\,i=Ne}) \end{matrix}\right\}_{t=2} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \left\{\begin{matrix} s(C_{P1\,i=1}) \\ s(C_{P2\,i=1}) \\ s(C_{P3\,i=1}) \\ s(C_{P4\,i=1}) \\ s(R_{INJ1\,i=1}) \\ s(S_{CONC\,i=1}) \end{matrix}\right\}_{t=TSTEP} & \left\{\begin{matrix} s(C_{P1\,i=2}) \\ s(C_{P2\,i=2}) \\ s(C_{P3\,i=2}) \\ s(C_{P4\,i=2}) \\ s(R_{INJ1\,i=2}) \\ s(S_{CONC\,i=2}) \end{matrix}\right\}_{t=TSTEP} & \cdots\cdots\cdots & \left\{\begin{matrix} s(C_{P1\,i=Ne}) \\ s(C_{P2\,i=Ne}) \\ s(C_{P3\,i=Ne}) \\ s(C_{P4\,i=Ne}) \\ s(R_{INJ1\,i=Ne}) \\ s(S_{CONC\,i=Ne}) \end{matrix}\right\}_{t=TSTEP} \\ NPV_{i=1} & NPV_{i=2} & \cdots\cdots\cdots & NPV_{i=Ne} \end{bmatrix} \qquad \textbf{(51)}$$

To do this standardization transformation, one has to define the maximum and minimum for each control. The advantage this transformation has is that it can set constraints to the controls through a standard normal distribution and the cumulative distribution function of each control. To impose constraints on the optimization process using the standard normal distribution the standardized control can be converted to a normal cumulative probability using the following equation for the standard normal cumulative distribution probability function (Jafarpour, 2008d):

$$\Psi[s(c)] = \frac{1}{2}\left[1 + erf\left(\frac{s(c)}{\sqrt{2}}\right)\right] \qquad \textbf{(52)}$$

where $\Psi[s(c)]$ refers to the probability of standardized controller *c* occurring in the cumulative standard normal curve. The probability $\Psi[s(c)]$ can then be converted to a control by using a cumulative uniform distribution which is a function of the minimum control, maximum control, and standard normal probability. This transformation from the standardized controller to the original control can be accomplished using the uniform distribution of controls in the following equation (Jafarpour, 2008d).

$$c = \begin{cases} c_{\min} & \text{for } \Psi[s(c)] = 0 \\ \Psi[s(c)] \cdot (c_{\max} - c_{\min}) + c_{\min} & \text{for } 0 < \Psi[s(c)] < 1 \\ c_{\max} & \text{for } \Psi[s(c)] = 1 \end{cases} \tag{53}$$

The obvious advantage of this transformation is that it constrains the controls to a maximum and minimum control by the use of the cumulative uniform distribution of controls throughout the optimization procedure. For example, if there is a controller *c* constrained by c_{min} and c_{max} that needs to be optimized the controller can be converted to the standard normal domain for the optimization and converted back to the controller domain using the standard normal cumulative distribution. The standard normal probability can then be used to determine the controller's value using the uniform cumulative distribution of the controller. This process can be seen in Figure 20 in which a new standardized controller z was obtained through the optimization process and converted from the standard normal domain to the controller domain using the uniform cumulative distribution curve of the control which is bounded by c_{min}=5 and c_{max}=20.

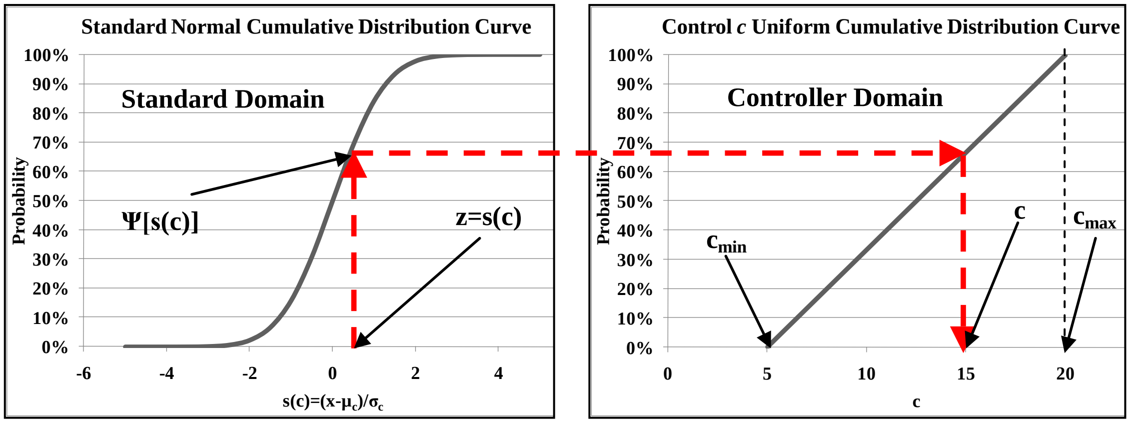


Figure 20: Normal Score Transformation to Uniform Set of Controls

The transformed matrix is the matrix used to advance the controllers for the next iteration, therefore the previously mentioned average Y matrix, C_y matrix, and y^{k+1} vector are based on the transformed matrix. The calculation of the transformed average Y matrix, transformed Cy matrix, and transformed y^{k+1} vector based on a transformed Y matrix can be seen in the following equation.

$$\overline{Y^*} = \frac{1}{Ne}\begin{bmatrix} \sum_{i=1}^{Ne} x_i^* \\ \sum_{i=1}^{Ne} \mathrm{NPV}_i \end{bmatrix} \tag{54}$$

$$C_y^* = \frac{1}{Ne-1}\left(Y^* - \overline{Y^*}\right)\left(Y^* - \overline{Y^*}\right)^T \tag{55}$$

$$y^{(k+1)^*} = \frac{1}{\alpha} C_y^* M^T + y_p^* \tag{56}$$

$$\begin{bmatrix} x \\ NPV \end{bmatrix}^{k+1} = \frac{1}{\alpha} C_y M^T + \begin{bmatrix} x \\ NPV \end{bmatrix}_p \tag{57}$$

The superscript, *, indicates that the matrix or vector is transformed. To use the controls in the simulator the transformed vector has to be changed back to the controller domain using the previously mentioned transformation procedures. The transformation of the Y ensemble matrix can be summed up in the following equation.

$$
\begin{gathered}
Y = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_{Ne} \\ \mathrm{NPV}_1 & \mathrm{NPV}_2 & \mathrm{NPV}_3 & \cdots & \mathrm{NPV}_{Ne} \end{bmatrix} \\
\Updownarrow TRANSFORM \\
Y^* = \begin{bmatrix} x_1^* & x_2^* & x_3^* & \cdots & x_{Ne}^* \\ \mathrm{NPV}_1 & \mathrm{NPV}_2 & \mathrm{NPV}_3 & \cdots & \mathrm{NPV}_{Ne} \end{bmatrix}
\end{gathered}
\qquad \textbf{(58)}
$$

The conversion from the controller domain, Y, to the transformed domain, Y^*, is used to perform the update of the controls with user defined controller constraints for every ensemble member. This transformation however, is done once in the beginning of the optimization routine on the original Y matrix (see optimization algorithm). The conversion from the transformed domain, Y^*, to the controller domain, Y, is used to run the simulator with the updated set of controls for every ensemble member (x_1 to x_{Ne}). This transformation has to be performed anytime a new Y^* is determined using the optimization routine.

6. DETERMINATION OF WEIGHTING FACTOR

The purpose of this section is to illustrate to the reader an alternative method of determining the weighting factor, alpha. This alternative method to Nwaozo's approach is not new but its application in determining the weighting factor is. As a result of this it is important to explain how this method is applied to this work.

The non-adjoint optimization method used in this research can be described as a steepest ascent method in which $C_Y M^T$ serves as the gradient and the weighting factor serves as the direction of the gradient. The key to this method is adequately finding the direction of the gradient that maximizes the NPV for the iteration efficiently. Nwaozo approach was inadequate because it required an adequate starting point of the weighting factor and step size. If the initial weighting factor and step size is inadequate it may take a very long time to search for the optimal alpha that adequately maximizes the NPV. To exacerbate this problem, if the initial weighting factor is not large enough one may never reach the optimal NPV because its optimal is much larger than the initial weighting factor. In addition to this problem, a large weighting factor may result in many optimization iterations while a small alpha may cause spurious results and overrun the controls out of their bounds and terminate the optimization iteration early. To address these problems a new approach is used to find the weighting factor.

In selecting a weighting factor algorithm there are certain properties the algorithm must have. The first property is that it must be able to approximately converge on an optimal alpha relatively quickly. The second property is that it must be able to shut off after a set number of iterations. The third property is that it must have a convergence error associated with it. The Gold-Section Search method (Chapra, et al., 2002) has all of these characteristics. This method is a simple general single variable search technique that relies on the concept of the golden ratio (Chapra, et al., 2002). The golden ratio, *R*, can be calculated in the following relation.

$$R = \frac{\sqrt{5}-1}{2} = .61803.... \quad \textbf{(59)}$$

The golden ratio is used to make the golden section search algorithm approach rapidly to an answer. The golden section search can be implemented by first choosing two extreme guesses $alpha_{low}$ and $alpha_{high}$ that bracket the optimal NPV or f(alpha). Two interior points, $alpha_1$ and $alpha_2$, can be calculated using the golden ratio.

$$d = R \cdot \left(alpha_{high} - alpha_{low}\right) \quad \textbf{(60)}$$

$$alpha_1 = alpha_{low} + d \quad \textbf{(61)}$$

$$alpha_2 = alpha_{high} - d \quad \textbf{(62)}$$

The simulator is then run to determine both f($alpah_1$) and f($alpha_2$). (Note f(alpha) consists of running the simulator for controls x_1 through x_{Ne} and averaging the NPV across each ensemble member). Two things may happen once this is done.

1. f($alpah_1$) > f($alpha_2$) then the domain of alpha to the left of $alpha_2$ from $alpha_{low}$ to $alpha_2$ can be eliminated because it does not contain the maximum. Then the intervals are redefined as the following:

$$alpha_{low} = alpha_{2,old} \quad \textbf{(63)}$$

$$alpha_2 = alpha_{1,old} \quad \textbf{(64)}$$

$$\mathrm{alpha}_1 = \mathrm{alpha}_{\mathrm{low}} + \mathrm{R} \cdot \left(\mathrm{alpha}_{\mathrm{high}} - \mathrm{alpha}_{\mathrm{low}}\right) \tag{65}$$

f($alpha_2$)= f($alpah_1$) and the f($alpah_1$) is determined by running the simulator.

2. f($alpah_2$) > f($alpha_1$) then the domain of alpha to the right of $alpha_1$ from $alpha_{high}$ to $alpha_1$ can be eliminated because it does not contain the maximum. Then the intervals are redefined as the following

$$\mathrm{alpha}_{\mathrm{high}} = \mathrm{alpha}_{1,\mathrm{old}} \tag{66}$$

$$\mathrm{alpha}_1 = \mathrm{alpha}_{2,\mathrm{old}} \tag{67}$$

$$\mathrm{alpha}_2 = \mathrm{alpha}_{\mathrm{high}} - \mathrm{R} \cdot \left(\mathrm{alpha}_{\mathrm{high}} - \mathrm{alpha}_{\mathrm{low}}\right) \tag{68}$$

f($alpha_1$)= f($alpah_2$) and the f($alpah_2$) is determined by running the simulator.

Once the first or second condition are satisfied then the new interval has be defined using the following possible conditions.

3. f($alpah_1$) > f($alpha_2$) then $alpha_{opt}$=$alpha_1$ and f($alpha_{opt}$)=f($alpha_1$).
4. f($alpah_2$) > f($alpha_1$) then $alpha_{opt}$=$alpha_2$ and f($alpha_{opt}$)=f($alpha_2$).

The conditions 1 through 4 can be repeated until convergence or until a minimum error is achieved which can be represented by the following expression.

$$\mathrm{e}_a = (1 - R) \cdot \left| \frac{\mathrm{alpha}_{\mathrm{high}} - \mathrm{alpha}_{\mathrm{low}}}{\mathrm{alpha}_{\mathrm{opt}}} \right| \cdot 100\% \tag{69}$$

This expression provides a condition in which the algorithm is terminated. The power of the golden section search method is that the interval containing the optimal alpha is reduced rapidly. In each iteration to find the optimal alpha the interval is reduced by a factor of the golden ratio or 61.8% (Chapra, et al., 2002). This concept can be seen in Table 9 and Table 10.

Table 9: Golden Section Search Interval of Convergence Parameters

alpha high=	1.00E+10
alpha low=	1.00E+03
R=	0.618033989

Table 10: Golden Section Search Interval of Convergence

Iteration	Interval as Percentage of Original Interval	Interval Length Containing Optimal
0	100	9999999000
5	9.016994375	901699347.3
10	0.813061876	81306179.43
15	0.073313744	7331373.625
20	0.006610696	661069.5474
25	0.000596086	59608.6039
30	5.3749E-05	5374.904461
35	4.84655E-06	484.6548329
40	4.37013E-07	43.70129902
45	3.94054E-08	3.940543675
50	3.55319E-09	0.355318601

It is very apparent the converging power of the golden section search method. In this work the maximum alpha, minimum alpha, maximum number of iteration until termination, and the maximum error allowed were 10^{10}, 1000, 100, and 50% respectively. The golden section search can be summarized in the following flow chart (Figure 21) for programming purposes.

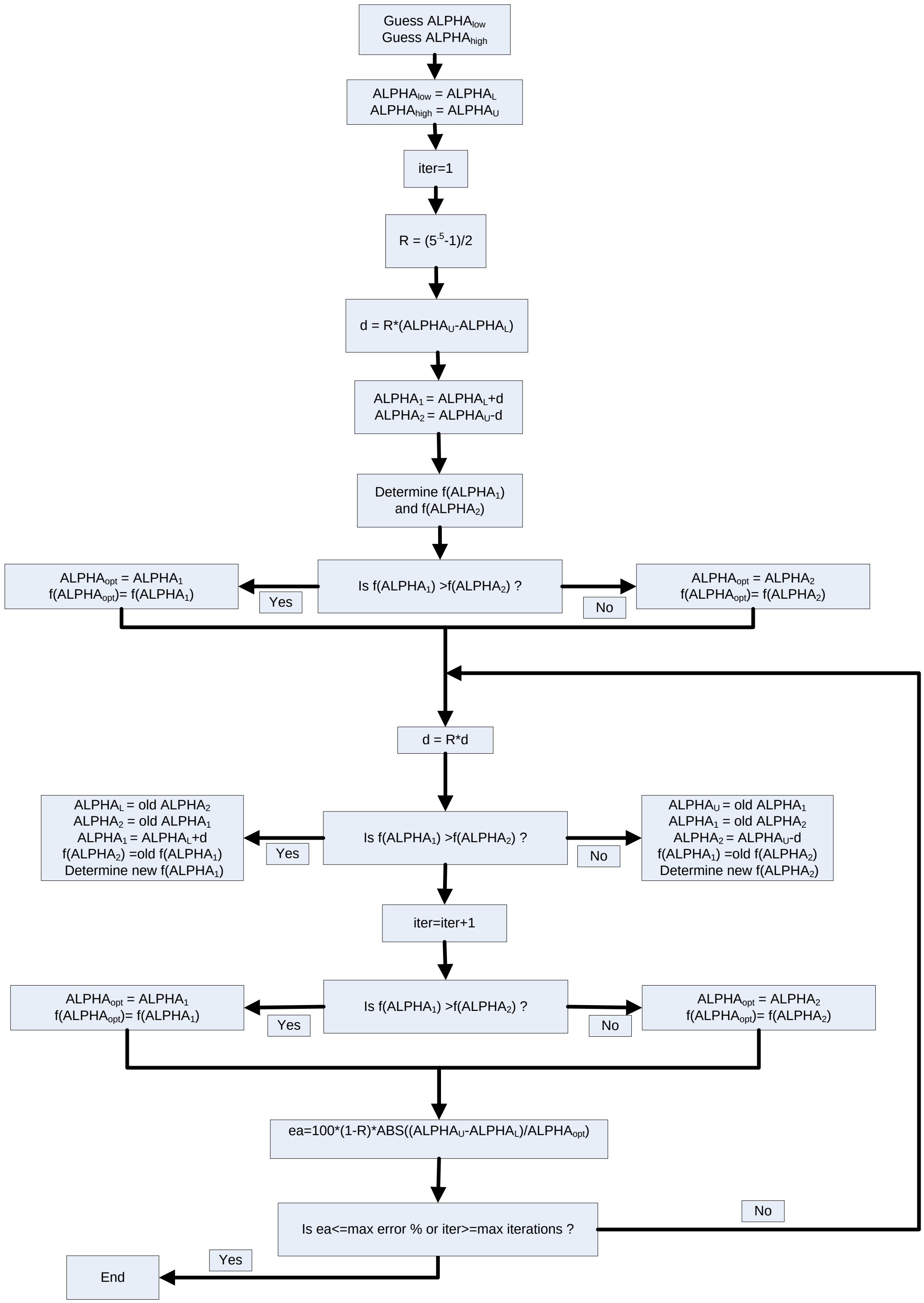


Figure 21: Golden Section Search Alpha Algorithm Flow Chart

7. ENGINEERING A PRIOR

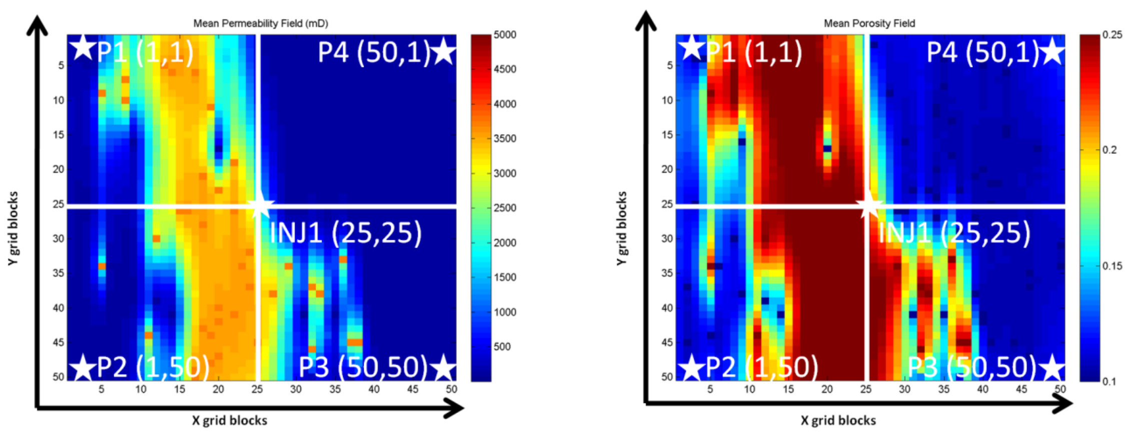
The purpose of this section is to illustrate to the reader the method in creating the several ensembles of the control vector *x*. Multiple ensembles of control vector *x* have to be created because the optimization method requires that a covariance matrix, C_y, be calculated. The covariance matrix represents the deviation between each ensemble of control x and the mean control x. Multiple ensembles of the control x cannot all be identical because it would mean a covariance matrix filled with zeroes. These zeroes would terminate the optimization prematurely (no gradient for the steepest ascent).

The design of the prior ensembles of the 5 spot surfactant flood field is important when using the Ensemble Kalman Filter optimization technique. When the Ensemble Kalman Filter is used as history match tool to match permeability to observations such as bottomhole pressure, the initial ensemble of permeability is routinely taken from a distribution of permeability based on empirical data in the field. As a corollary to history matching, the distribution of the ensemble of initial controls should also be based on physical laws in petroleum engineering. This work addresses these concerns in the design of the initial prior for the Ensemble Kalman Filter optimization process for the total liquid production rate control of the producer wells scenario and bottomhole pressure control of the producer wells scenario. Both scenarios also include the control of liquid injection rate and control of injected surfactant concentration.

A few assumptions were made concerning the design of the initial prior. These assumptions are the following:

1. Transient response when the well first opens and also when the well changes controls. This assumption covers the response in which the wells have not felt the pressure boundaries yet due to non boundary dominated flow.
2. Ideal design is when the reservoir pressure is close to the initial reservoir pressure at all times during production.
3. Controls are held constant from one user given time interval until the next time interval. Therefore the controls are essentially step wise.

For the design of the controls using these assumptions, the field was delineated into four regions. This was done to assign each region a production well for the 5 spot pattern in the 320 acre heterogeneous field (see Figure 22).



(a) (b)

Figure 22: Field Regions for Permeability and Porosity: (a) Permeability Field, (b) Porosity Field

Each region contains a characteristic average permeability, k, and porosity, Φ, corresponding to the production wells region. For example wells P1, P2, P3, and P4 refer to regions 1, 2, 3, and 4. Therefore region 1 characteristic average porosity and average permeability are referred to as k_1 and Φ_1.

7.1 TOTAL LIQUID PRODUCTION RATE CONTROL

The total liquid production rate control is based on conventional decline curve analysis and the assumption that for this case of control the well has not felt its boundaries. Decline curve analysis attempts to find the best fit curve that matches the production data and forecasts this curve for future economic production. But to serve the purpose of having a covariance matrix for the optimization method, decline curve analysis was used to generate multiple realizations of the liquid rate. The decline curve type that was used in this research was the hyperbolic decline cure. This decline curve was chosen because changing the liquid production rate to different values during the production life implies that the well never feels its boundaries and thus exhibits transient behavior. For this research only the beginning of the time interval was available for the control (control is constant between different time intervals). Therefore the hyperbolic decline equation (Mian, 2002a) was altered to account for this control. The modified liquid control case equation is the following:

$$q(t) = Q_{init} \cdot \left(1 + \frac{b \cdot D_{init} \cdot (t-1) \cdot DT}{365}\right)^{-1/b} \qquad \textbf{(70)}$$

The Q_{init} stands for the initial production rate at the start of the production, D_{init} stands for the initial decline rate at the start of the production, t stands for the time interval through the production life, DT stands for the length of the time interval, and b stands for a factor that describes whether the well is in boundary or non boundary dominated flow. Values of b from 0 to 1 are associated with boundary dominated flow while values of b greater than 1 are associated with non boundary dominated flow or transient well behavior.

Each well in the 5 spot flood has a defined controlled liquid rate. The equations describing the controls for each production well can be seen here.

$$q_1(t) = Q_{init} \cdot \left(1 + \frac{b \cdot D_{init} \cdot (t-1) \cdot DT}{365}\right)^{-1/b} \quad \textbf{(71)}$$

$$q_2(t) = Q_{init} \cdot \left(1 + \frac{b \cdot D_{init} \cdot (t-1) \cdot DT}{365}\right)^{-1/b} \quad \textbf{(72)}$$

$$q_3(t) = Q_{init} \cdot \left(1 + \frac{b \cdot D_{init} \cdot (t-1) \cdot DT}{365}\right)^{-1/b} \quad \textbf{(73)}$$

$$q_4(t) = Q_{init} \cdot \left(1 + \frac{b \cdot D_{init} \cdot (t-1) \cdot DT}{365}\right)^{-1/b} \quad \textbf{(74)}$$

Typically when using decline curve analysis to curve fit production history; the Q_{init}, D_{init}, and b terms are fit used using a least square analysis. But for this research; the Q_{init}, D_{init}, and b terms were varied by first creating realizations of these parameters and then sampling from the distribution of these parameters to create realizations of hyperbolic decline throughout the production time. The following equations illustrate the equations for uniform distribution of realizations of Q_{init}, D_{init}, and b.

$$Q^{i}_{init} = RN \cdot \left(Q^{Max}_{init} - Q^{Min}_{init}\right) + Q^{Min}_{init} \quad \textbf{(75)}$$

$$D^{i}_{init} = RN \cdot \left(D^{Max}_{init} - D^{Min}_{init}\right) + D^{Min}_{init} \quad \textbf{(76)}$$

$$b_i = RN \cdot \left(b_{Max} - b_{Min}\right) + b_{Min} \quad \textbf{(77)}$$

The Max superscript indicates the maximum for the parameter, RN stands for a random number between 0 and 1, Min stands for minimum, and i indicates the ith realization. The preceding set of equations was used to generate realizations for wells P1, P2, P3, and P4. An example set of the realizations of liquid flow rate can be seen by first defining minimum and maximum parameters for Q_{init}, D_{init}, and b (see Table 11):

Table 11: Example Qinit, b, and Dinit Parameters

	mean	minimum	max
Q_{init}, STB/DAY	500	100	900
b	5	1	9
D_{init}, /year	5	0.1	9.9

Using this table of parameters 40 realizations of Qinit, Dinit, and b can be seen in Figure 23, Figure 24, and Figure 25.

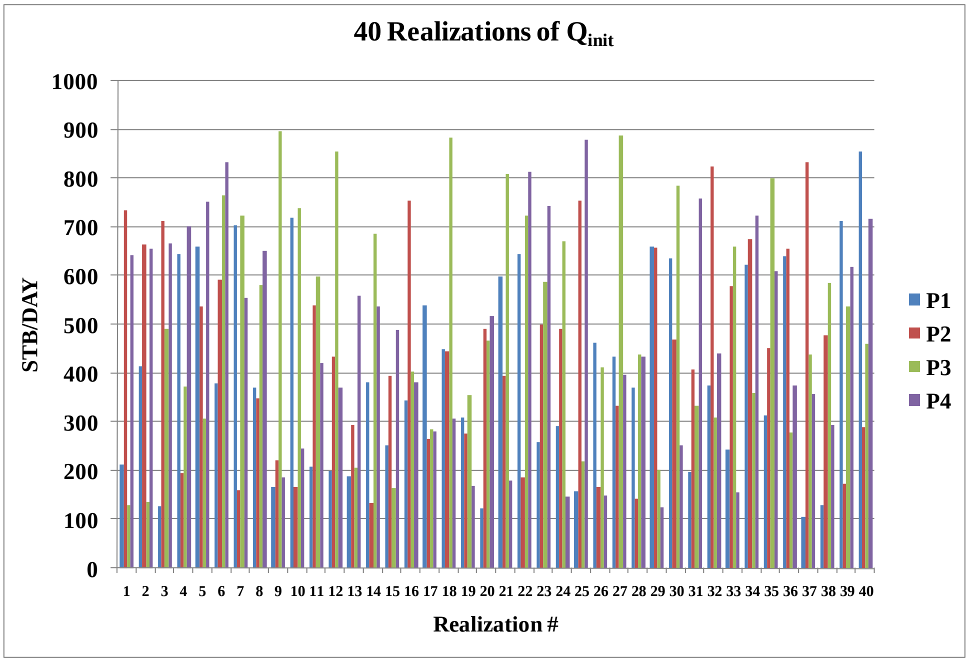


Figure 23: 40 Realizations of Q_{init}

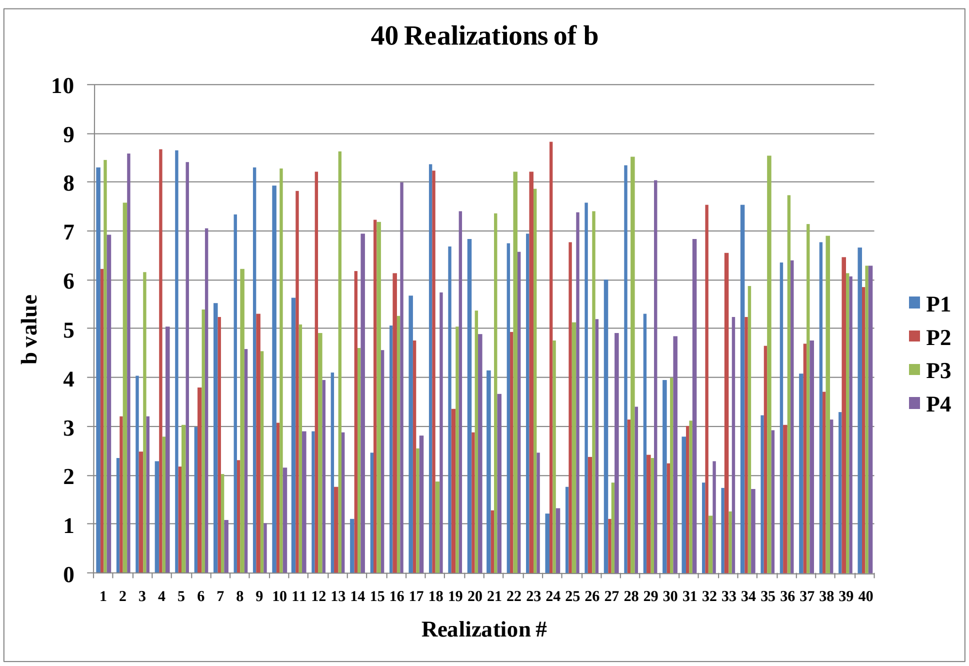


Figure 24: 40 Realizations of b

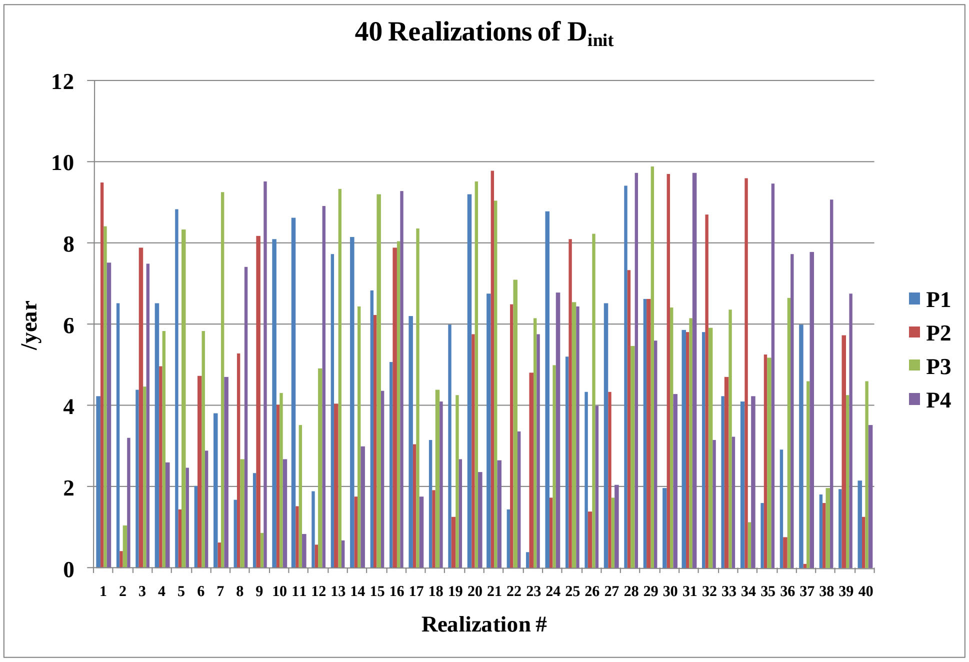


Figure 25: 40 Realizations of D_{init}

These realizations of parameters were used to generate 40 realizations of liquid flow rate controls for a 10 year time period with 6 month control change (20 time intervals). The first realization of liquid flow rate control for all four production wells can be seen in Figure 26. 40 realizations of liquid control for each well can be seen in Figure 27.

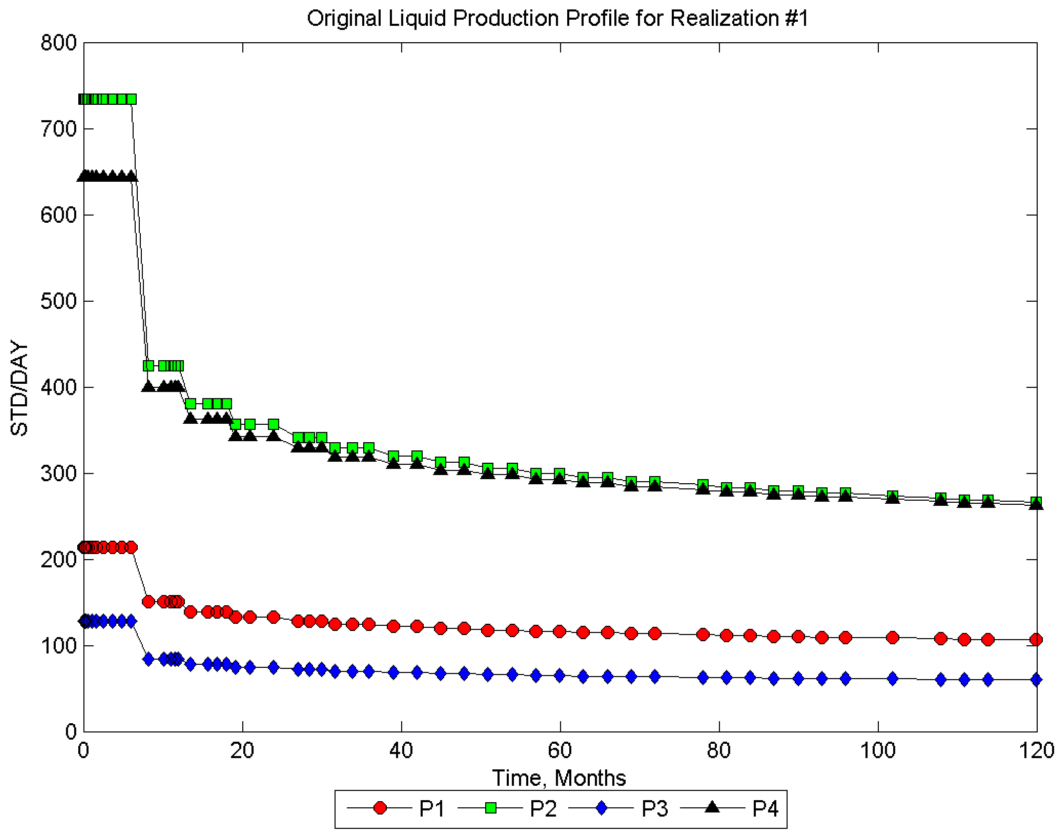


Figure 26: 1st Realization of Liquid Flow Rate Control Prior Example

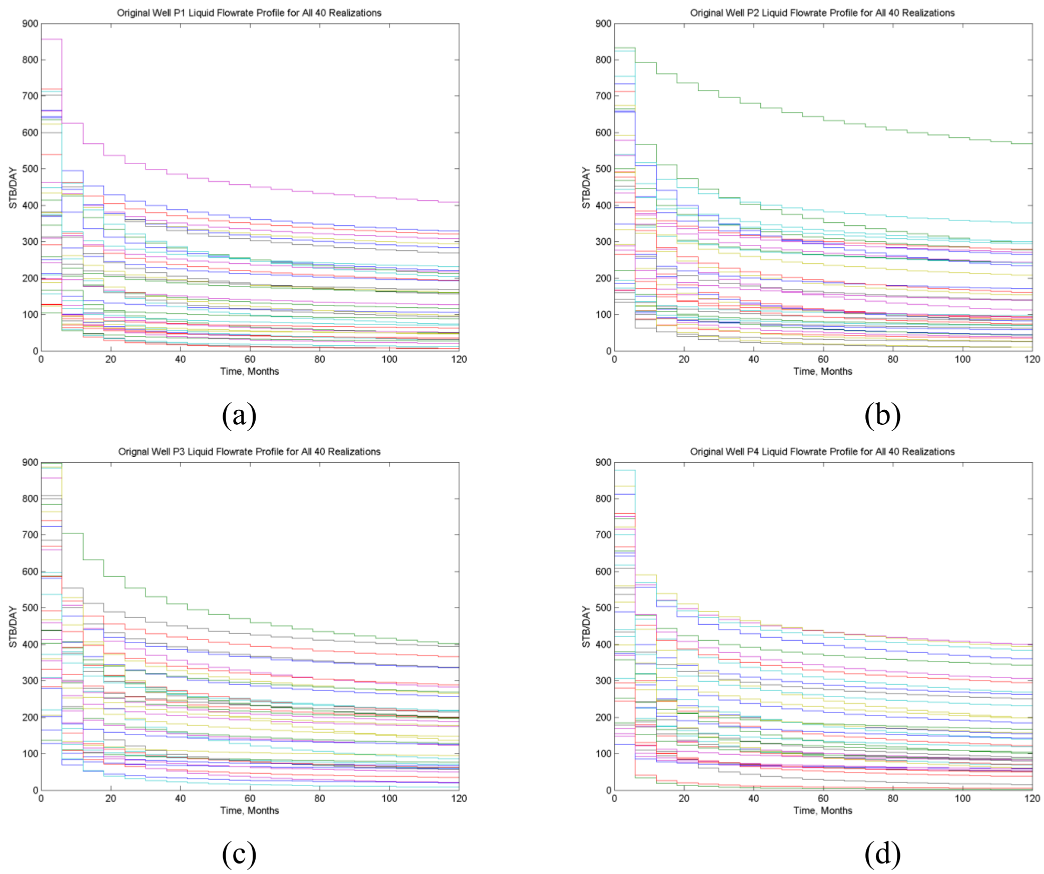


Figure 27: Realizations of Total Liquid Flow Rate for Producer Wells: (a) Well P1, (b) Well P2, (c) Well P3, (d) Well P4

7.2 BOTTOMHOLE PRESSURE CONTROL

As mentioned earlier, changing the controls prevents the wells from feeling the boundaries. As a result of this, the bottomhole pressure control was based on the transient inflow performance relationship. This relationship can be seen here in the following expression (Economides, et al., 1994).

$$q(t)=\frac{kh(P_i-BHP)}{162.6B\mu}\left(\log(DT\cdot t\cdot 24)+\log\left(\frac{k}{\phi\mu c_t r_w^{\ 2}}\right)-3.23\right)^{-1} \quad \textbf{(78)}$$

The P_i indicates the initial reservoir pressure, BHP stands for the bottomhole flowing pressure, B stands for the formation volume factor, μ stands for the viscosity, h stands for the net pay, c_t stands for the total compressibility, and r_w stands for the inner radius of the well. To use the transient flow equation, the transient productivity index was derived from the transient inflow performance relationship. This relation can be seen here were the J stands for the productivity index.

$$J = \frac{q}{(P_i - BHP)} = \frac{kh}{162.6B\mu}\left(\log(DT \cdot t \cdot 24) + \log\left(\frac{k}{\phi\mu c_t r_w^{\ 2}}\right) - 3.23\right)^{-1} \quad \textbf{(79)}$$

This productivity index was modified to account for the stepwise change in controls that occur in this research. For this research the productivity index was altered to account for control at the beginning of the timestep and the first hour of change. This change can be seen in the following equations.

$$J = \frac{q}{(P_i - BHP)} = \frac{kh}{162.6B\mu}\left(\log(t_e) + \log\left(\frac{k}{\phi\mu c_t r_w^{\ 2}}\right) - 3.23\right)^{-1} \quad \textbf{(80)}$$

$$t_e = DT \cdot (t-1) \cdot 24 + 1 \quad \textbf{(81)}$$

In the reservoir there are primarily two phases flowing at any time thus the total productivity index was split into a water phase and oil phase productivity index. This phase breakdown can be seen in the following expression.

$$J_T = J_o + J_w \quad \textbf{(82)}$$

Each region then was assigned a productivity index according to the properties of that region. Expansion of the total productivity index and region property delineation can be seen in the following expressions of the productivity index for region 1, 2, 3, and 4.

$$J_{T,1}(t) = \frac{k_1 \cdot k_{ro,1} \cdot h}{162.6 B_o \mu_o} \left(\log(t_e) + \log\left(\frac{k_1 \cdot k_{ro,1}}{\phi_1 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} +$$

$$\frac{k_1 \cdot k_{rw,1} \cdot h}{162.6 B_w \mu_w} \left(\log(t_e) + \log\left(\frac{k_1 \cdot k_{rw,1}}{\phi_1 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} \tag{83}$$

$$J_{T,2}(t) = \frac{k_2 \cdot k_{ro,2} \cdot h}{162.6 B_o \mu_o} \left(\log(t_e) + \log\left(\frac{k_2 \cdot k_{ro,2}}{\phi_2 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} +$$

$$\frac{k_2 \cdot k_{rw,2} \cdot h}{162.6 B_w \mu_w} \left(\log(t_e) + \log\left(\frac{k_2 \cdot k_{rw,2}}{\phi_2 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} \tag{84}$$

$$J_{T,3}(t) = \frac{k_3 \cdot k_{ro,3} \cdot h}{162.6 B_o \mu_o} \left(\log(t_e) + \log\left(\frac{k_3 \cdot k_{ro,3}}{\phi_3 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} +$$

$$\frac{k_3 \cdot k_{rw,3} \cdot h}{162.6 B_w \mu_w} \left(\log(t_e) + \log\left(\frac{k_3 \cdot k_{rw,3}}{\phi_3 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} \tag{85}$$

$$J_{T,4}(t) = \frac{k_4 \cdot k_{ro,4} \cdot h}{162.6 B_o \mu_o} \left(\log(t_e) + \log\left(\frac{k_4 \cdot k_{ro,4}}{\phi_4 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} +$$

$$\frac{k_4 \cdot k_{rw,4} \cdot h}{162.6 B_w \mu_w} \left(\log(t_e) + \log\left(\frac{k_4 \cdot k_{rw,4}}{\phi_4 \mu c_t r_w^{\ 2}} \right) - 3.23 \right)^{-1} \tag{86}$$

The productivity index that was used was at the first 1 hour of control change for the respective region. The following expressions describe the bottomhole pressure controls for each region based on the productivity index of the region and the hyperbolic decline flow rate of each well.

$$BHP_1(t) = P_i - \frac{q_1(t)}{J_{T,1}(t)} \quad \textbf{(87)}$$

$$BHP_2(t) = P_i - \frac{q_2(t)}{J_{T,2}(t)} \quad \textbf{(88)}$$

$$BHP_3(t) = P_i - \frac{q_3(t)}{J_{T,3}(t)} \quad \textbf{(89)}$$

$$BHP_4(t) = P_i - \frac{q_4(t)}{J_{T,4}(t)} \quad \textbf{(90)}$$

An example using these controls for a 10 year time period with 6 month control change (20 time intervals) can be seen in Figure 28 for the 1st realization. 40 realizations of bottomhole pressure control are illustrated in Figure 29.

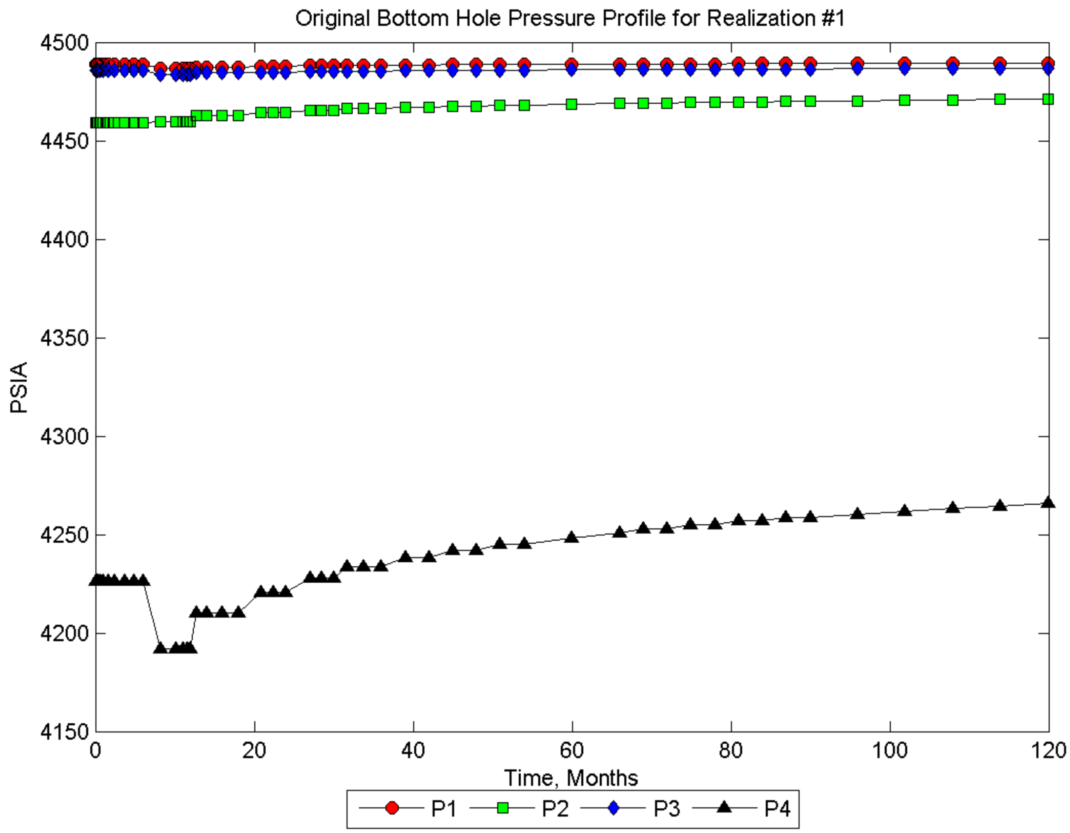
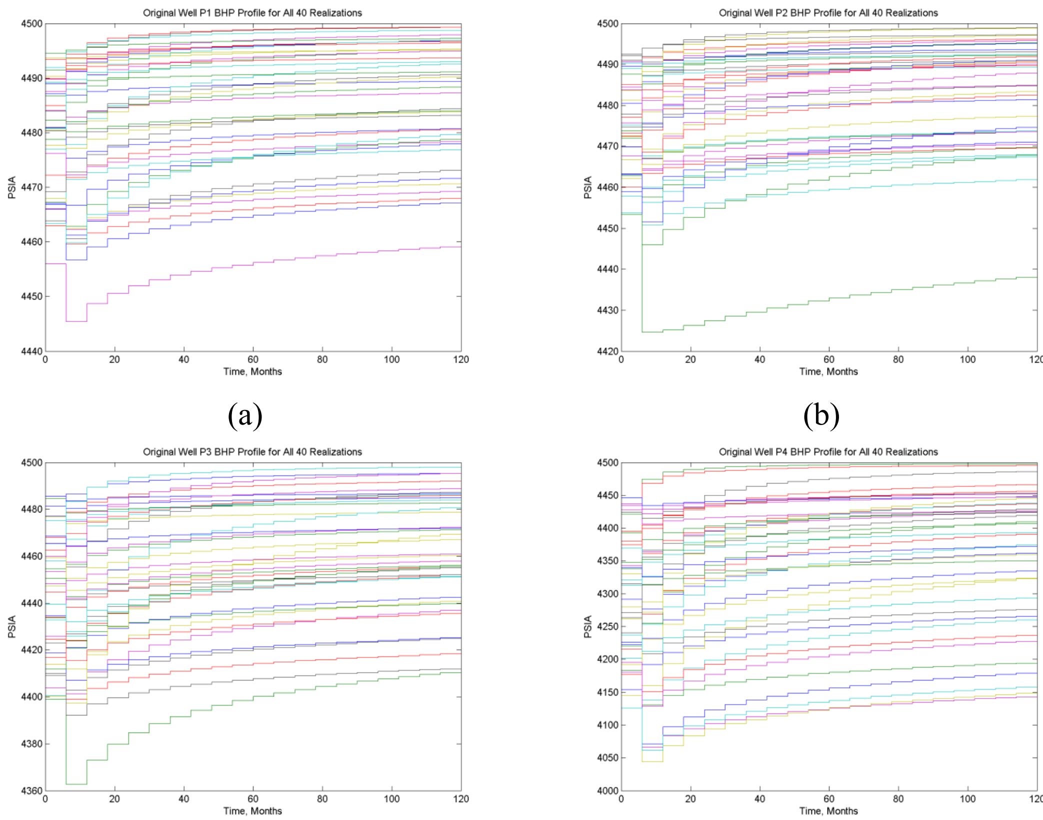


Figure 28:1st Realization of Bottomhole Pressure Control Prior Example



(a) (b)

(c) (d)

Figure 29: Realizations of Bottomhole Pressure for Producer Wells: (a) Well P1, (b) Well P2, (c) Well P3, (d) Well P4

The preceding figures were generated using the hyperbolic total flow rate mentioned earlier and the parameters from the Reservoir Description section.

7.3 INJECTION RATE CONTROL

The initial injection rate control was designed to maintain the reservoir pressure as close to the initial reservoir pressure as possible. This was done by re-injecting the same volume that is produced. The injection rate control for the liquid control option can be seen in the following expressions.

$$q_1(t) + q_2(t) + q_3(t) + q_4(t) = q_T(t) \quad \textbf{(91)}$$

$$R_{INJ1}(t) = q_T(t) \quad \textbf{(92)}$$

An example of this control for a 10 year time period with 6 month control change (20 time intervals) can be seen in Figure 30 for the 1st realization. 40 realizations of this control are illustrated in Figure 31.

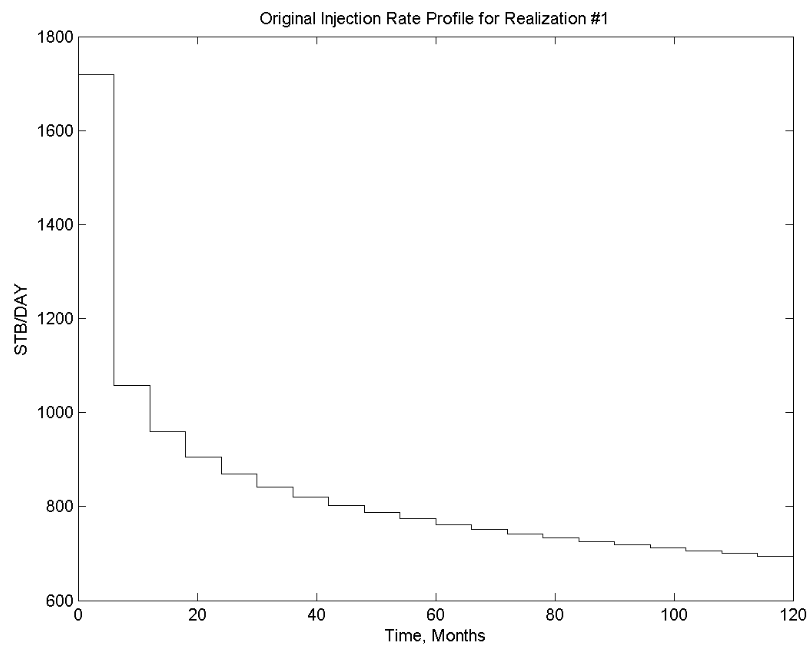


Figure 30:1st Realization of Total Flowrate Control Prior Example (Liquid Case)

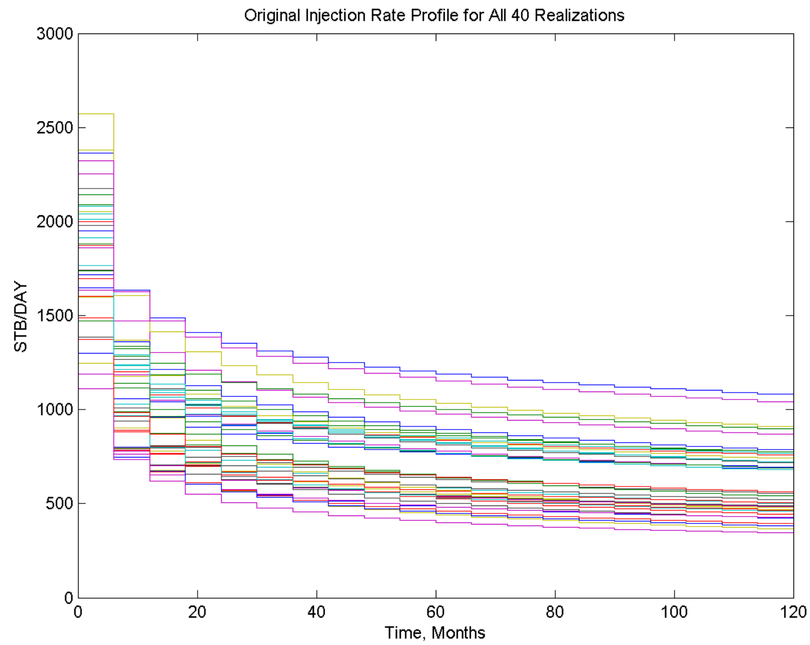


Figure 31:40 Realizations of Total Flowrate Control (Liquid Case)

The preceding figures were generated using the total liquid flow rate control example mentioned earlier.

The injection rate control for the bottomhole pressure control is similar to the total liquid flow rate control. The flow rate for each region was put in terms of the bottomhole pressure control parameters. The flow rate from each region was then summed up to form the equation for the total injection rate. This equation can be seen here.

$$R_{INJ1}(t) = \left(P_i - BHP_1(t)\right)\cdot J_{T,1}(t) + \left(P_i - BHP_2(t)\right)\cdot J_{T,2}(t) + \left(P_i - BHP_3(t)\right)\cdot J_{T,3}(t) + \left(P_i - BHP_4(t)\right)\cdot J_{T,4}(t) \quad \textbf{(93)}$$

An example of this control for a 10 year time period with 6 month control change (20 time intervals) can be seen in Figure 32 for the 1st realization. 40 realizations of this control are conveyed in Figure 33.

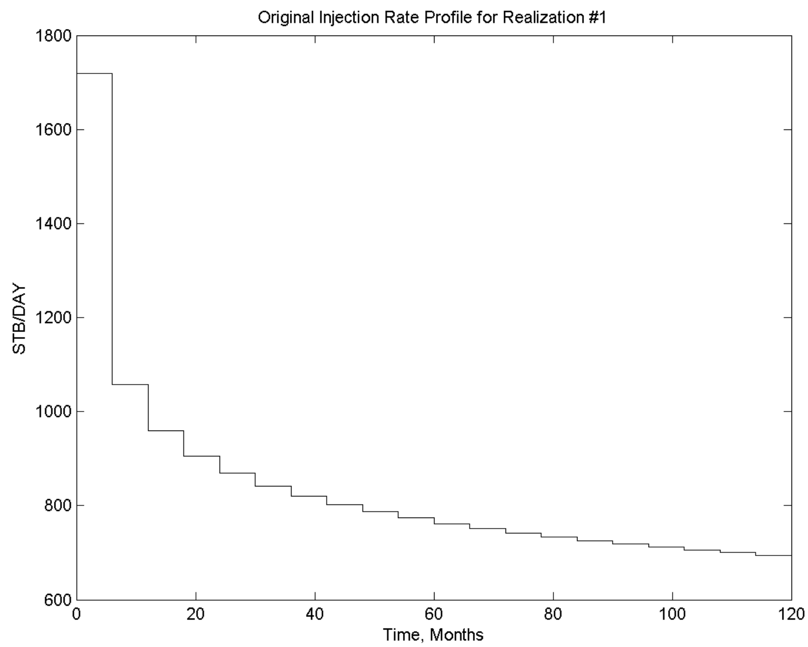


Figure 32:1st Realization of Total Flowrate Control Prior Example (BHP Case)

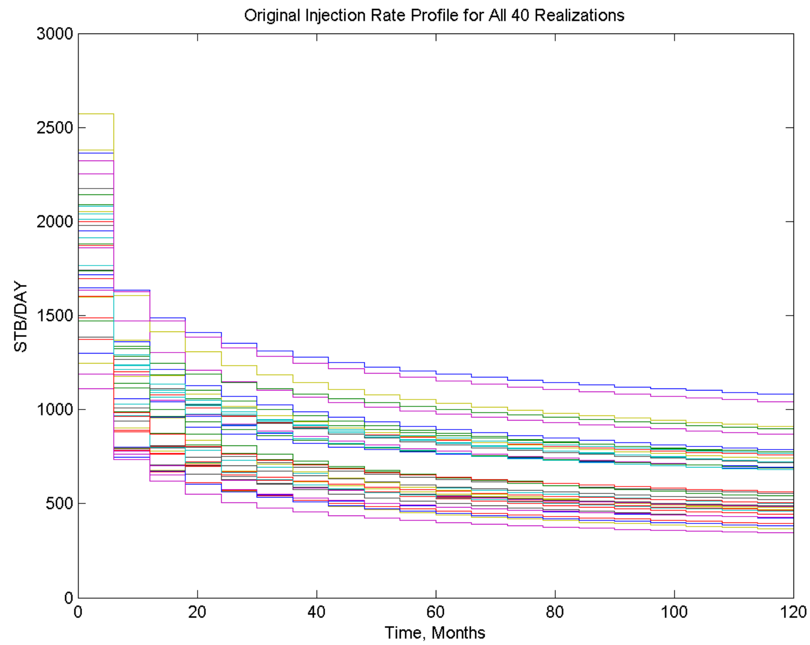


Figure 33:40 Realizations of Total Flowrate Control Prior (BHP Case)

The preceding figures were generated using the bottomhole pressure control example mentioned earlier.

7.4 SURFACTANT CONCENTRATION CONTROL

The surfactant concentration control design was based on injecting the surfactant concentration at a constant value through the life of the production. This was done because the actual mass of surfactant injected varies with time because the injection rate of water varies with time. The concentration to inject the concentration of surfactant was based on the lowest concentration of surfactant that lowered the interfacial tension to the highest degree. This value based on the surfactant properties is .35053 lb/STB.

8. OPTIMIZATION IMPLEMENTATION

The optimization routine was a large part of the general program written in this research. Before this routine is explained, it is important to know the procedure for running control vector, x, and calculating the resulting NPV based on this control vector.

To write the control vector, x, it is important to know the position of each control element. The elements in control vector, x, are arranged in the following sequence from t=1 to t=TSTEP.

$$\{C_{P1}, C_{P2}, C_{P3}, C_{P4}, R_{INJ1}, S_{CONC}\}_{t=1}, \ldots, \{C_{P1}, C_{P2}, C_{P3}, C_{P4}, R_{INJ1}, S_{CONC}\}_{t=TSTEP} \tag{94}$$

In each time interval there are 6 controls. This fact can be used to describe the position of each element in the control vector, x. For example, the position for S_{CONC} for timestep t is 6t. Similarly, for C_{P1} the position is 6t-5. The position of S_{CONC} for time interval t in the control vector, x, can be written as x[6t]. Using this logic and nomenclature the position of each element in the control vector, x, can be written as the following sequence from t=1 to t=TSTEP.

$$\{x[6t-5], x[6t-4], x[6t-3], x[6t-2], x[6t-1], x[6t]\}_{t=1}, \ldots$$
$$\{x[6t-5], x[6t-4], x[6t-3], x[6t-2], x[6t-1], x[6t]\}_{t=TSTEP} \tag{95}$$

Knowing the position of each element in control vector, x, makes it possible for the controls to be written in Matlab and used in Eclipse. Figure 34 illustrates the procedure for running the i^{th} ensemble member of control vector, x_i, to the simulator and calculating the NPV associated with the ensemble member of control vector, x_i.

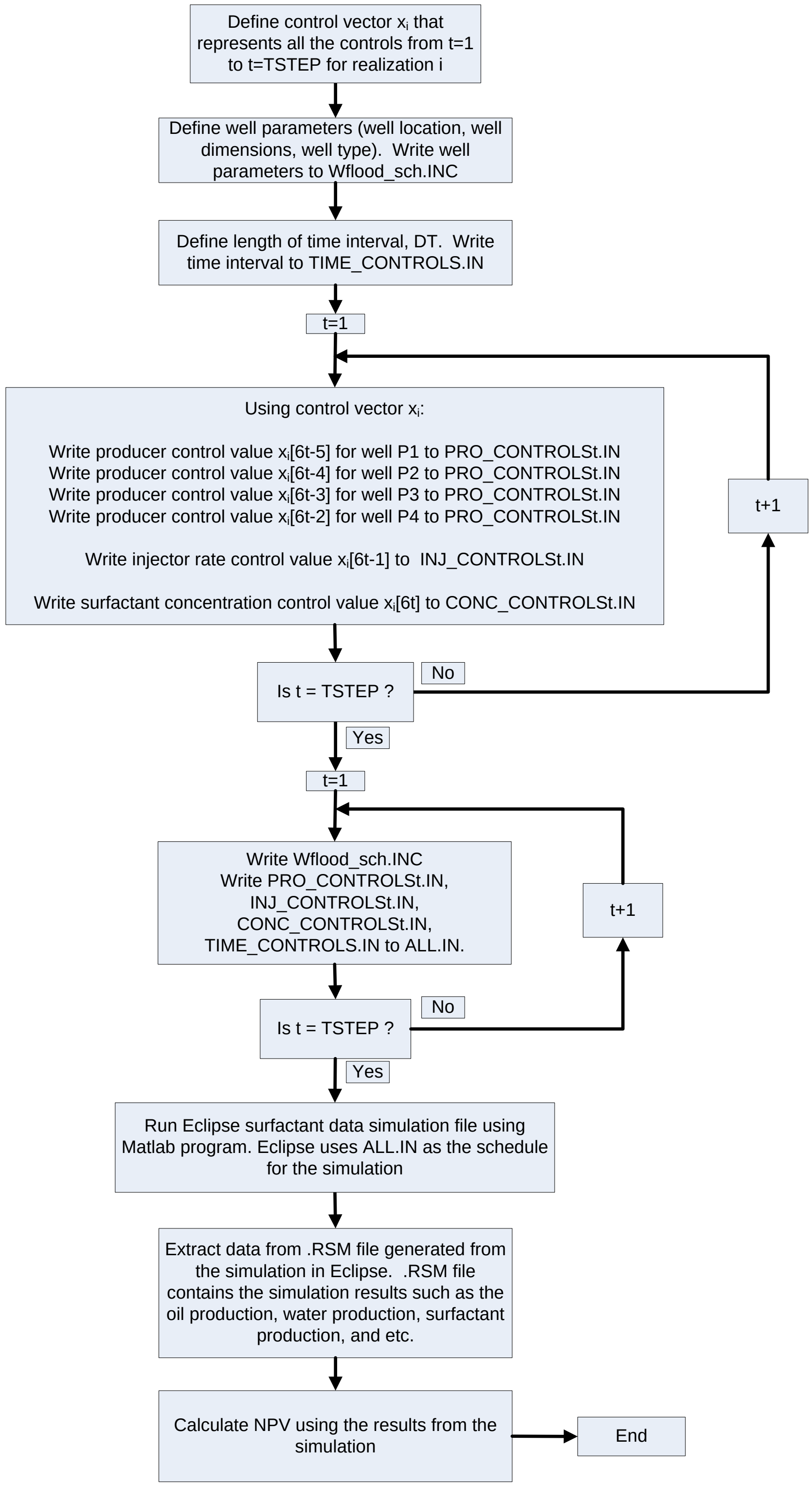


Figure 34: Matlab Program to Run Control Vector x_i Using Eclipse

The previous outlined procedure is the general routine used to communicate between Matlab and Eclipse. The file that links them together is the ALL.IN file. This file contains all the values in the control vector, x. Matlab creates this file, while Eclipse uses this file as the schedule for the controls. Eclipse activates the ALL.IN file in the simulation file by using the INCLUDE keyword followed by ALL.IN. An example ALL.IN file for 5 time intervals of control (t=5) is displayed in Figure 35. The ALL.IN file contains the files for the well parameters (Wflood_sch.INC), production controls (PRO_CONTROLSt.IN), injection rate controls (INJ_CONTROLSt.IN), surfactant concentration controls (CONC_CONTROLSt.IN), and length of time interval (TIME_CONTROLS.IN). Examples of these files are illustrated in Figure 36 (example is for time interval t=1 with liquid rate control of the producer wells).

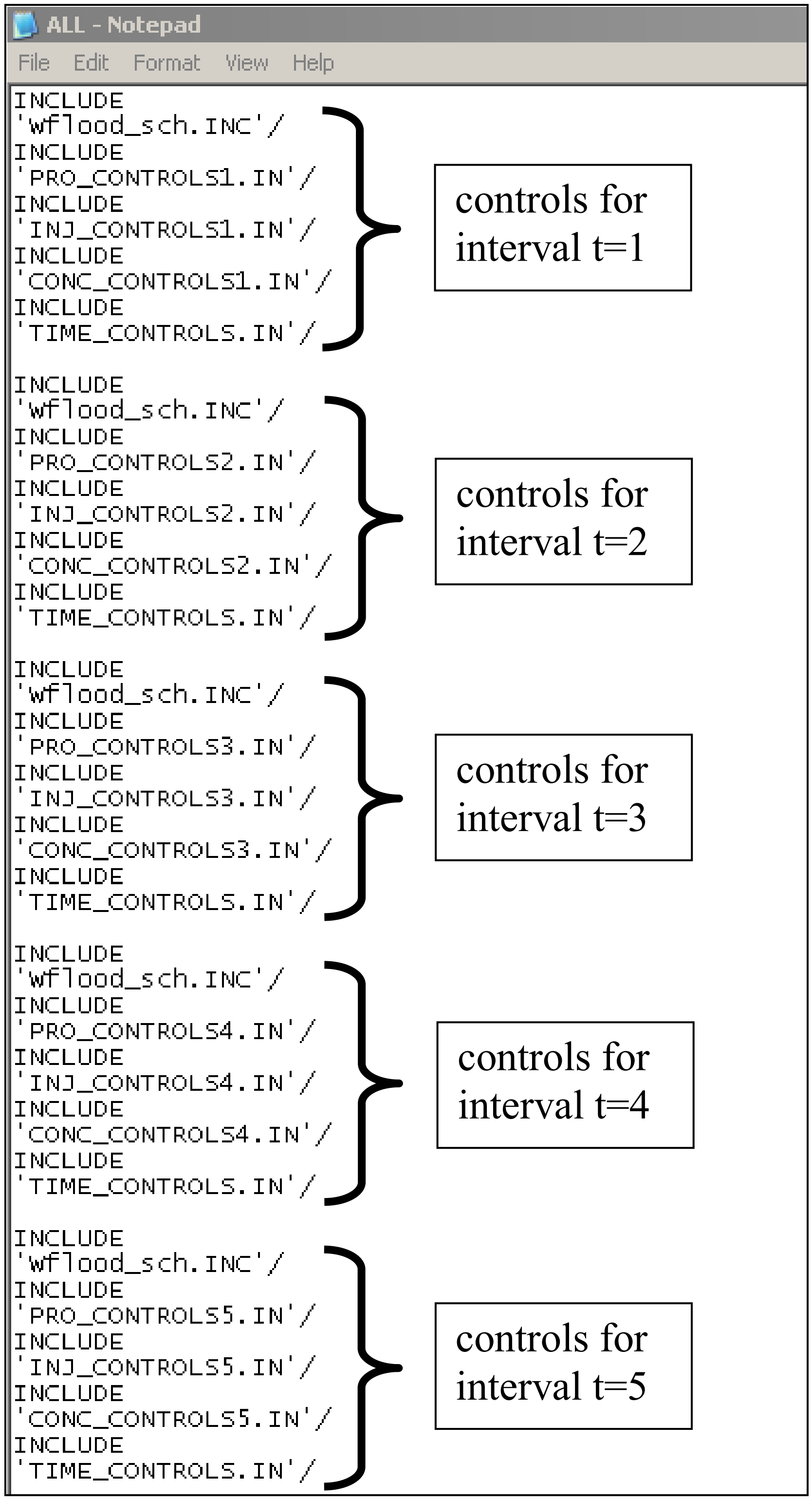
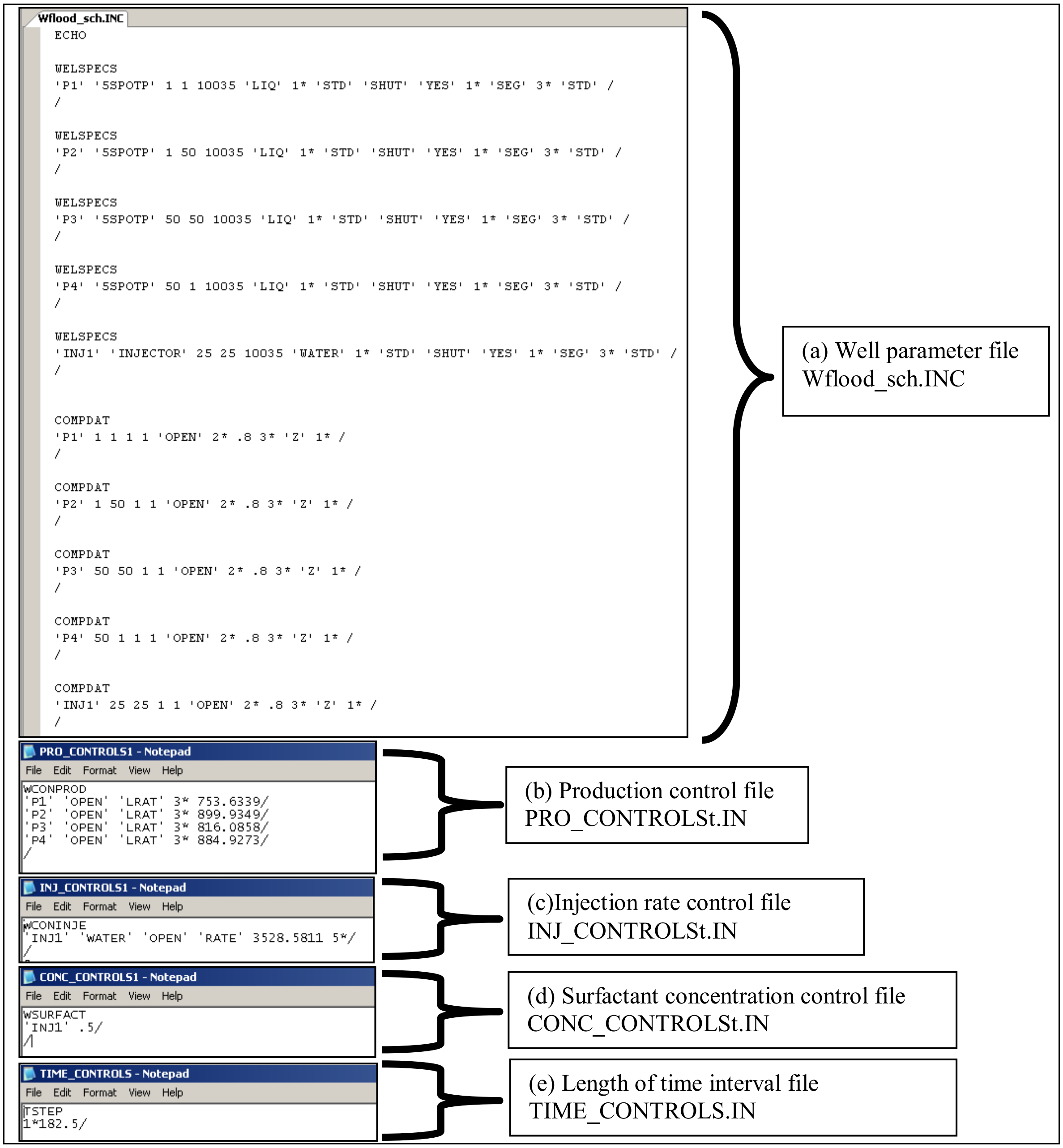


Figure 35: Example ALL.IN File for t=5



```
Wflood_sch.INC
ECHO

WELSPECS
'P1' '5SPOTP' 1 1 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'P2' '5SPOTP' 1 50 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'P3' '5SPOTP' 50 50 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'P4' '5SPOTP' 50 1 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'INJ1' 'INJECTOR' 25 25 10035 'WATER' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/


COMPDAT
'P1' 1 1 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'P2' 1 50 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'P3' 50 50 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'P4' 50 1 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'INJ1' 25 25 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/
```



```
PRO_CONTROLS1 - Notepad
File Edit Format View Help
WCONPROD
'P1' 'OPEN' 'LRAT' 3* 753.6339/
'P2' 'OPEN' 'LRAT' 3* 899.9349/
'P3' 'OPEN' 'LRAT' 3* 816.0858/
'P4' 'OPEN' 'LRAT' 3* 884.9273/
/
```



```
INJ_CONTROLS1 - Notepad
File Edit Format View Help
WCONINJE
'INJ1' 'WATER' 'OPEN' 'RATE' 3528.5811 5*/
/
```



```
CONC_CONTROLS1 - Notepad
File Edit Format View Help
WSURFACT
'INJ1' .5/
/
```



```
TIME_CONTROLS - Notepad
File Edit Format View Help
TSTEP
1*182.5/
```


Figure 36: Matlab Files Written for Eclipse Schedule File ALL.IN: (a) Well Parameter File, (b) Production Control File, (c) Injection Rate Control File, (d) Surfactant Concentration Control File, (e) Length of Time Interval File

The previous example files were formatted for use by Eclipse. The routine that created the files that linked Matlab and Eclipse were programmed to write the control vector, x, for any time interval length, number of time intervals, and producer control type.

The optimization routine generates the initial prior ensemble of controls, transforms them, calculates the appropriate weighting factor for each non-adjoint optimization, and performs the actual non-adjoint optimization. The general overview of this optimization process can be seen in Figure 37. The optimization process terminates when the average NPV for all of the ensemble members has stopped increasing.

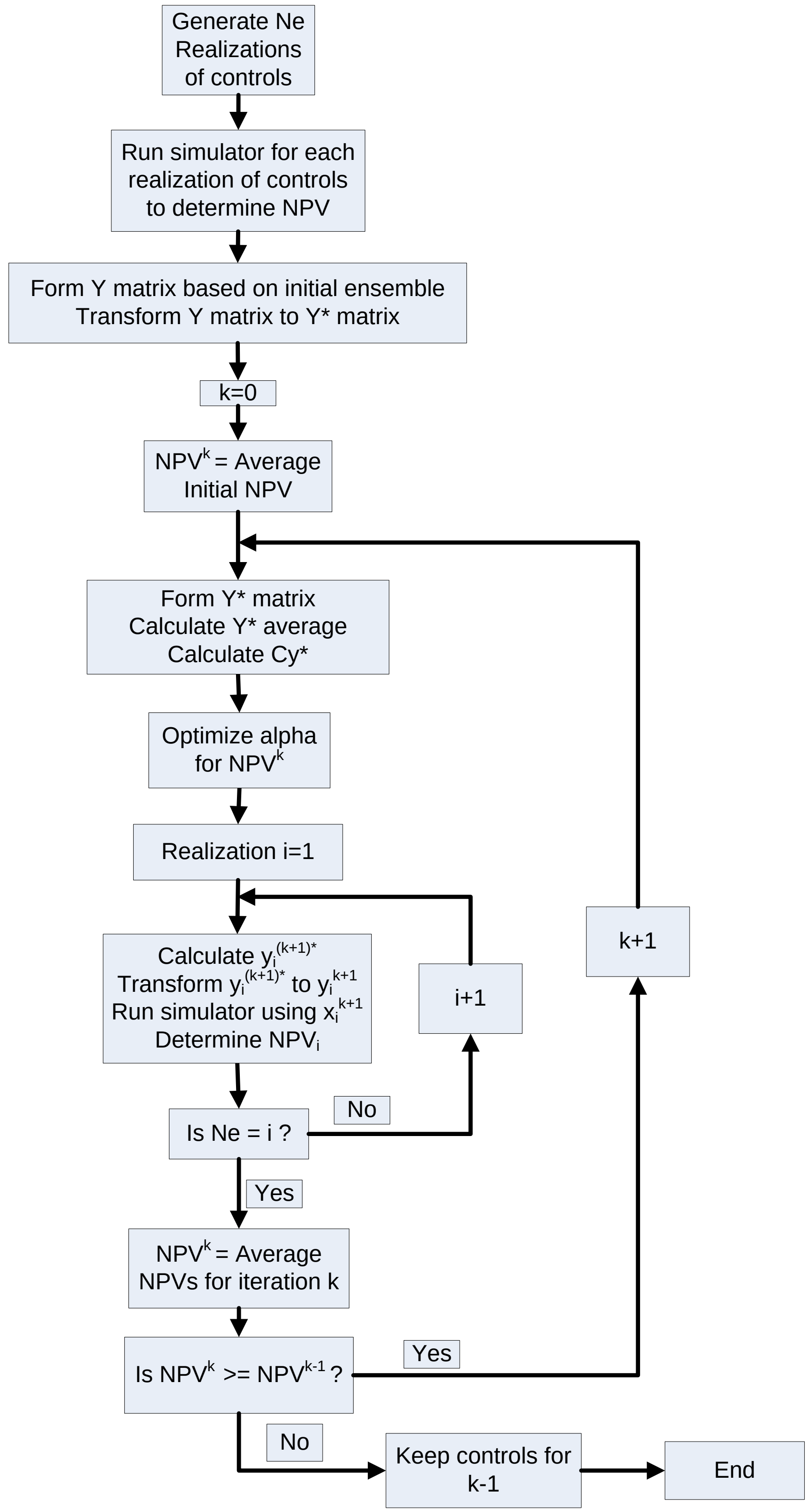


Figure 37: Optimization Program Flow Chart

9. ECONOMICS

The NPV calculation was critical in the determination of the optimal control parameters for the 5 spot surfactant flood field. To calculate NPV the following relations were used (Nwaozo, 2006):

$$\mathrm{NPV} = \sum_{v=1}^{V} \frac{C_v}{\left(1 + \frac{r}{365}\right)^{t_v}} \quad \textbf{(96)}$$

$$C_v = \left(\mathrm{OPR} \cdot \$_{\mathrm{OIL}} - \mathrm{WPR} \cdot \$_{\mathrm{Water}} - \mathrm{WINJ} \cdot \$_{\mathrm{WaterINJ}} - \mathrm{SURF} \cdot \$_{\mathrm{Surfactant}}\right) \cdot \Delta t_v \quad \textbf{(97)}$$

where the C_v refers to the cash flow for the time step (as reported by the simulator) v, r is the discount rate percentage, t_v is the time in days corresponding to time step v, Δt_v is the change in time step from v-1 to v in days, OPR is the oil production rate in standard barrels per day, WPR is the water production rate in standard barrels per day, WINJ is the water injection rate in standard barrels per day, SURF is the surfactant injection rate in lbs per day, $\$_{\mathrm{OIL}}$ is the selling price of oil in US dollar per barrel, $\$_{\mathrm{Water}}$ is the cost of treating the water in US dollar per barrel, $\$_{\mathrm{WaterINJ}}$ is the cost of the injected water in US dollar per barrel, and $\$_{\mathrm{Surfactant}}$ is the cost of the surfactant in US dollar per pound of surfactant.

NPV is a measure of a project's success. A positive NPV indicates that the project more than pays for its own cost while a negative NPV indicates that the project does not generate enough income to sustain itself financially (Mian, 2002a). It is important to note that the economic parameters oil price, surfactant cost, water injection cost, and water treatment cost are set constant for one NPV calculation. This is a reasonable assumption because the NPV accounts for the interest earned during the project (Mian, 2002a). Therefore, it is technically incorrect to change the economic parameters during the life of the project if the NPV is used only to screen the success of the project during its life.

9.1 ECONOMIC REALIZATIONS

Once the optimal solution was obtained, several realizations of the economic parameters (oil price, surfactant cost, water injection cost, and water treatment cost) were used along with realizations of the permeability field to account for every possible scenario. To do this, a triangular distribution of the economic parameters was used to create possible realizations of the economic parameters. Triangular distribution requires a most likely value or mode (Tr_M), a minimum value (Tr_L), and a maximum value(Tr_H). The triangular distribution equation depends on the magnitude of a random number, RN, which is between 0 and 1. For different cases of the random number, the triangular distribution (Mian, 2002b) is the following equation for an economic parameter Tr.

$$\mathrm{Tr} = \begin{cases} \mathrm{Tr_L} + \sqrt{(\mathrm{Tr_M} - \mathrm{Tr_L})\cdot(\mathrm{Tr_H} - \mathrm{Tr_L})\cdot \mathrm{RN}} & \text{for RN} \leq \dfrac{(\mathrm{Tr_M} - \mathrm{Tr_L})}{(\mathrm{Tr_H} - \mathrm{Tr_L})} \\ \mathrm{Tr_H} - \sqrt{(\mathrm{Tr_H} - \mathrm{Tr_M})\cdot(\mathrm{Tr_H} - \mathrm{Tr_L})\cdot(1 - \mathrm{RN})} & \text{for RN} \geq \dfrac{(\mathrm{Tr_M} - \mathrm{Tr_L})}{(\mathrm{Tr_H} - \mathrm{Tr_L})} \end{cases} \tag{98}$$

Table 12: Surfactant Flood Economic Parameters

	Mode	Minimum	Maximum
oil price, $/STB	70	20	120
water handeling, $/STB	2	1	3
water injection, $/STB	1	0.7	2
surfactant price, $/lb	1	0.8	1.5

Table 12 contains the economic values used in this work. The discount rate percentage, r, was set constant at 10%. The optimization routine uses the mode economic parameters while the Monte Carlo sampling uses the realizations of all the economic parameters which are based on the triangular distribution.

10. MONTE CARLO OF GEOLOGICAL AND ECONOMIC REALIZATIONS

Once the optimization was completed using the mean permeability field, mean porosity field, and the mode of the oil price; realizations of, permeability, and economic parameters were used to generate a picture of the probability of success using the optimal control. This was done by sampling every possible scenario of permeability field and economic parameters. This process can be understood by envisioning R_k realizations of permeability fields, R_{op} realizations of oil price, R_{wi} realizations of injection cost, R_{wt} realizations of water treatment cost, and R_{surf} realizations of surfactant cost. The sampling process can then be thought of by first generating the first scenario which is the 1st realization of permeability field, 1st realization of oil price, 1st realization of water injection cost, 1st realization of water treatment cost, and 1st realization of surfactant cost. After sampling of the first scenario the second scenario would then be the 1st realization of permeability field, 1st realization of oil price, 1st realization of water injection cost, 1st realization of water treatment cost, and 2nd realization of surfactant cost. This process would then be repeated until all scenarios are sampled for all realizations of permeability and economic parameters. If all possible realizations are sampled then there would be $R_kR_{op}R_{wi}R_{wt}R_{surf}$ number of possible scenarios. Once the Monte Carlo sampling was finished, the cumulative probability plot of the NPV was created using a cumulative distribution plot (preprogrammed in Matlab).

The cumulative probability plot of NPV has on its vertical axis cumulative probability and NPV on its horizontal axis. It is important to understand the interpretation of this plot. If at the mode economic parameters the NPV is positive at a corresponding probability, P_{mode}, but at probabilities less than P_{mode} the NPV is negative it just means that the project is not successful for economic scenarios worse than the mode economic parameters. The cumulative probability plot of NPV is a measure of the project's success for given range of economic climate at the start of the project. The project success should not be taken as a failure if the NPV is shown to be negative for cumulative probabilities less than P_{mode} because the project has shown success using the mode economic scenarios.

11. PROGRAM IMPLEMENTATION

The program written in Matlab for this research performs a non-adjoint optimization of the mean of NPV of the ensembles using the mean of the realizations of permeability field, mean of the realizations of porosity field, and mode of the economic parameters. The program then uses the optimized 1st ensemble of the control vector to perform Monte Carlo sampling on the realizations of permeability field and economic parameters. Once this is finished a program written in Matlab generates the graphics needed for analysis. Figure 38 illustrates the flow chart of the entire program written in Matlab.

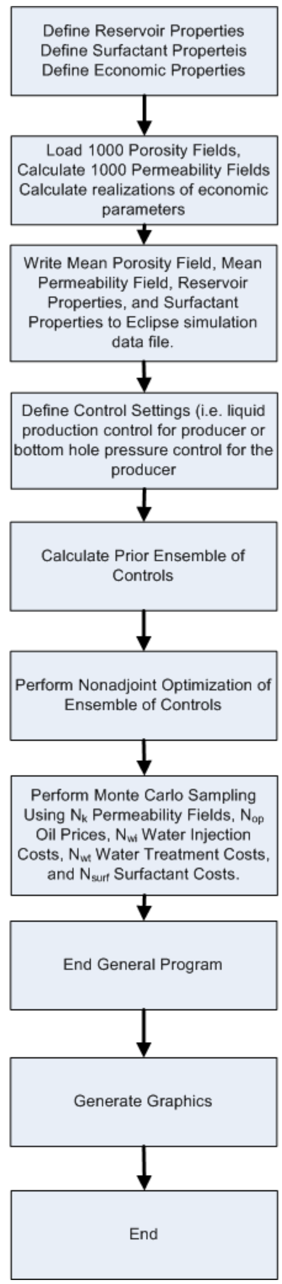


Figure 38: Overall Program Outline

The optimization and incorporation of uncertainty in the economics and permeability field requires many simulation runs. To generate the initial prior ensemble matrix, Y, it takes Ne simulation runs in Eclipse. To determine the proper weighting factor it takes at least 3Ne simulation runs. To perform the optimization, it takes at least two optimization runs. To perform the Monte Carlo simulation, it takes R_k actual simulation runs and $R_k R_{op} R_{wi} R_{wt} R_{surf}$ calculations of NPV. The graphics section of the program requires 2Ne simulation runs to get the observations using the original and optimized control for each ensemble member of control. The time it takes to run one simulation is approximately 9 seconds. The time it takes to calculate 100 NPV (for Monte Carlo) is approximately 9 seconds. In total the necessary time it takes for the entire program written in this research to finish can be expressed in the following equation.

$$\boldsymbol{Req.Time(hours)} \geq \frac{9\cdot[Ne+2\cdot(3\cdot Ne)+2\cdot Ne+R_k]+\frac{9}{100}\cdot[R_k\cdot R_{op}\cdot R_{wi}\cdot R_{wt}\cdot R_{surf}]}{60\cdot 60} \quad \textbf{(99)}$$

The computer used was an Intel Core 2 Quad CPU with 2.66 GHz for each core and 3.25 GB of RAM. The "Req. Time (hours)" equation can be used to plan required optimization time using the CPU for this research.

12. RESULTS AND ANALYSIS

The purpose of this section is to investigate the implications of using a water flood compared to a surfactant flood, to investigate the consequences of loosely constrained controls and tightly constrained controls, and to also investigate how project time length affects the economics of EOR. There are advantages and disadvantages of using a water flood or a surfactant flood. A water flood may be cheaper to handle when compared to a surfactant flood due to the cost of surfactants. However, a water flood cannot reduce the oil saturation below the residual oil saturation; while the surfactant flood can because of its ability to reduce the interfacial tension. Optimization of NPV using both methods may answer the question which is the more economical flood. A study of the consequences of loosely constrained controls versus tightly constrained controls can give insight into choosing the constraints of the controls for optimization. An investigation of the length of time of the project may give lessons of when to use a surfactant flood or waterflood. To investigate the goals of this section, two time scales were analyzed. The first time scale is production for 20 years with a 6 month control change. In this time scale the controls were loosely constrained and the producer control type was bottomhole pressure. The second time scale is production for 40 years with a 6 month control change. In this time scale the controls were tightly constrained and the producer control type was liquid rate. In both time scales the initial water saturation at the start of the project was set at 40%, the number of ensemble members was 40 (Ne=40), and both the water and surfactant flood cases were investigated. For the probabilistic sampling, 10 realizations each of the water treatment cost, water injection cost, surfactant cost, oil price, and permeability fields were used for each case. These realizations correspond to 10^5 scenarios for each case.

Because the average porosity varies spatially throughout the grid, a summation approach to the original oil in place calculation had to be used to calculate the original oil in place. The following equation was used to determine the original oil in place:

$$OOIP = \frac{7758 \cdot (1 - S_w) \cdot h}{B_o} \sum_{BX=1}^{GBX} \sum_{BY=1}^{GBY} \bar{\phi}_{BX,BY} \cdot A \qquad \textbf{(100)}$$

where BX corresponds to the grid block in the horizontal direction; BY, to the grid block in the vertical direction; A, to the area in acres; GBX, to the total number of grid blocks in the horizontal direction; and GBY, to the total number of grid blocks in the vertical direction. The original oil in place using the water saturation and the mean porosity field is illustrated in Figure 39.

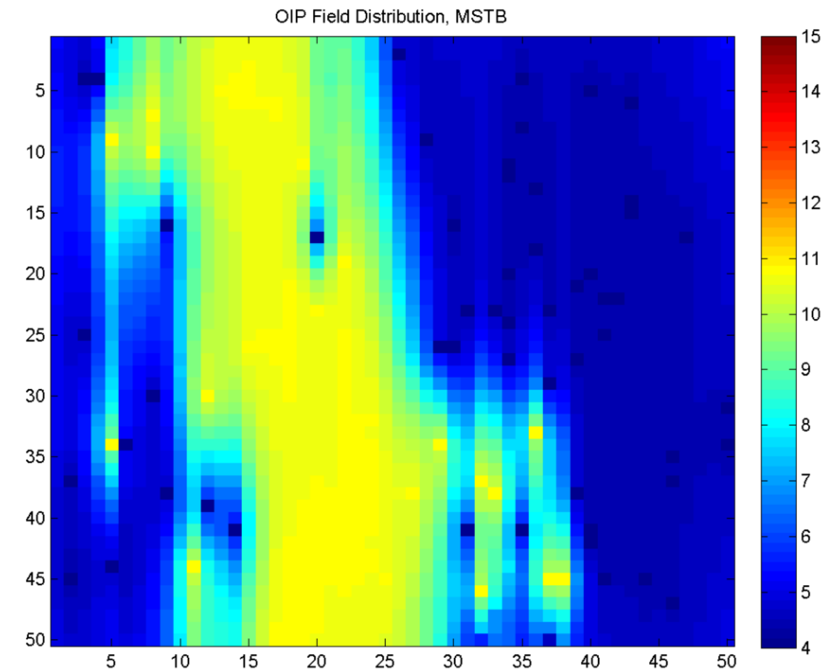


Figure 39: Original Oil in Place Field Distribution

The Original Oil in Place was calculated to be approximately 17.5MMSTB for all cases.

12.1 PRODUCTION TIME OF 20 YEARS WITH BOTTOMHOLE PRESSURE CONTROL OF THE PRODUCER

Production time of 20 years with bottomhole pressure control of the producer was analyzed with loosely defined constraints. Loosely defined constraints correspond to the situation in which the controls for the producer and injector are allowed to have a wide range of values. The purpose of studying loosely defined controls is to show the consequences of having constraints that represent a wide range of values during optimization. Table 13 shows the DCA parameters for this scenario. Table 14 shows the constraints for this scenario.

Table 13: DCA Parameters for Production Time of 20 Years

	mean	minimum	max
Q_{init}, STB/DAY	1000	100	1900
b	5	1	9
D_{init}, /year	5	0.1	9.9

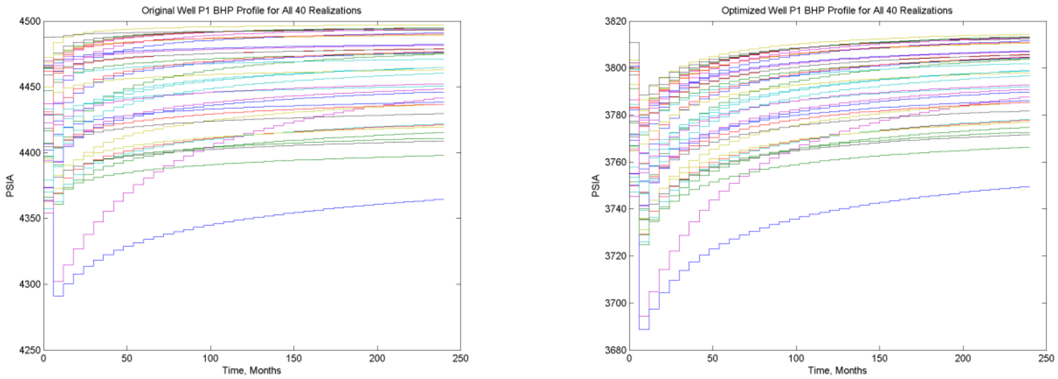
Table 14: Constraints for Production Time of 20 Years

	Minimum	Maximum	Mean	Standard Deviation
Bottom Hole Pressure, PSIA	50	4500	2275	2225
Injection Rate, STB/DAY	40	4000	2020	1980
Surfactant Concentration, LB/STB	0.01	10	5.005	4.995

These DCA parameters and constraints were used to study the optimization of a water flood and surfactant flood for a production time of 20 years with a 6 month control change.

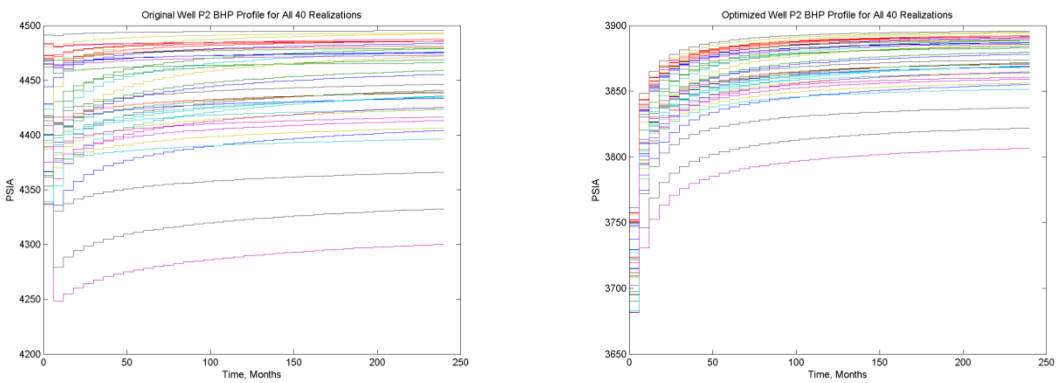
12.1.1 WATER FLOOD FOR 20 YEARS WITH BOTTOM HOLE PRESSURE CONTROL OF THE PRODUCER

Using the defined original DCA parameters and constraints for the production time of 20 years; Figure 40, Figure 41, Figure 42, Figure 43, and Figure 44 illustrate the original and optimized controls for a water flood for all ensemble members.



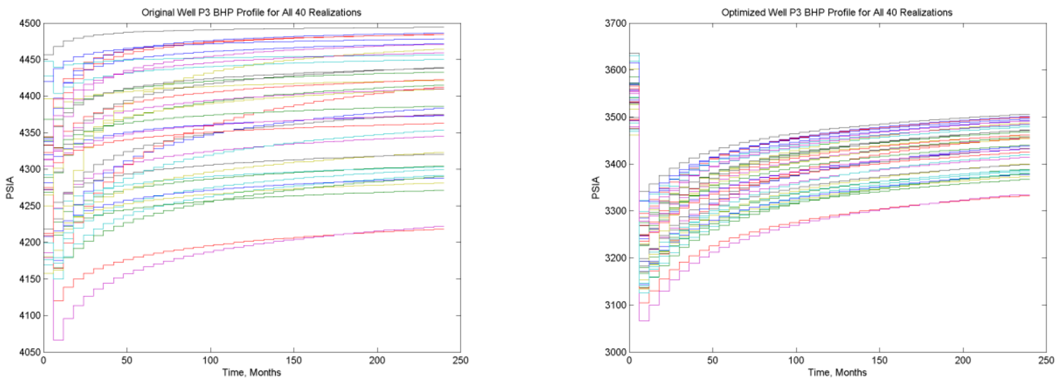
(a) (b)

Figure 40: Well P1 Controls for Water Flood for 20 Years: (a) Original, (b) Optimized



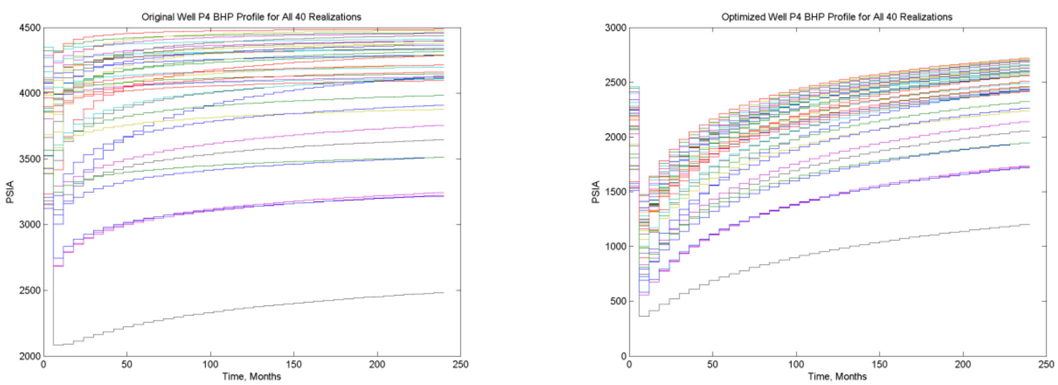
(a) (b)

Figure 41: Well P2 Controls for Water Flood for 20 Years: (a) Original, (b) Optimized



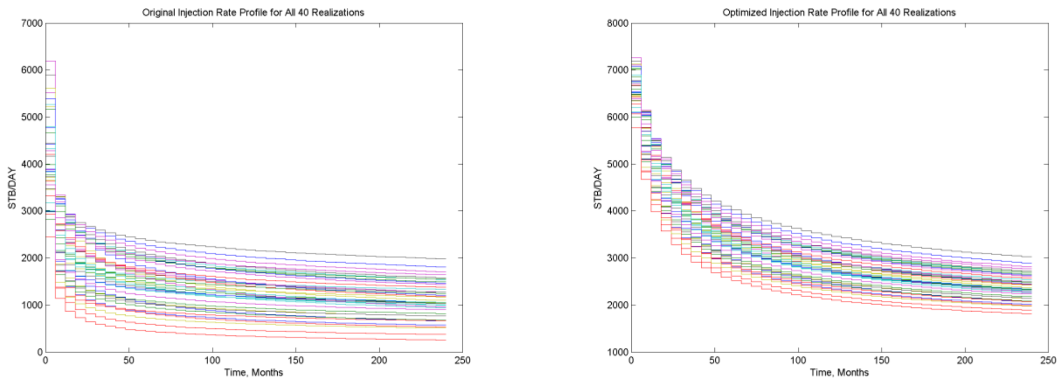
(a) (b)

Figure 42: Well P3 Controls for Water Flood for 20 Years: (a) Original, (b) Optimized



(a) (b)

Figure 43: Well P4 Controls for Water Flood for 20 Years: (a) Original, (b) Optimized



(a) (b)

Figure 44: Injection Rate Controls for Water Flood for 20 Years: (a) Original, (b) Optimized

It can be seen that the trends in the original controls are maintained in the optimized controls. Another observation that can be inferred is that the variability among the realization of the controls is reduced.

Figure 45 and Figure 46 show the resultant NPV and cumulative oil distribution obtained from optimizing the process for all ensemble members.

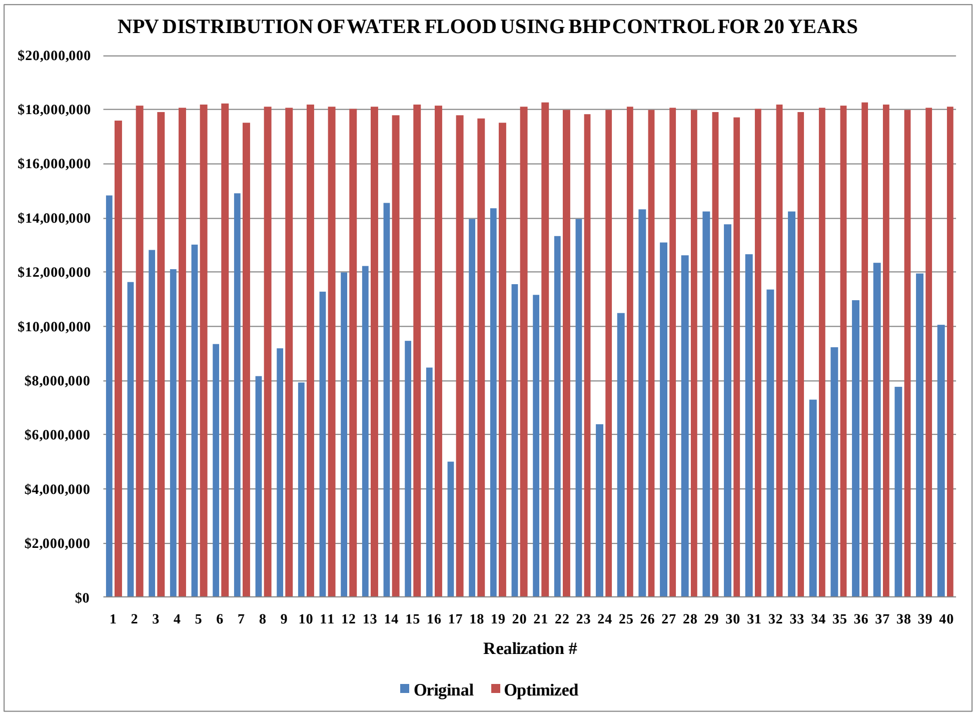


Figure 45: NPV Distribution of Water Flood for 20 Years

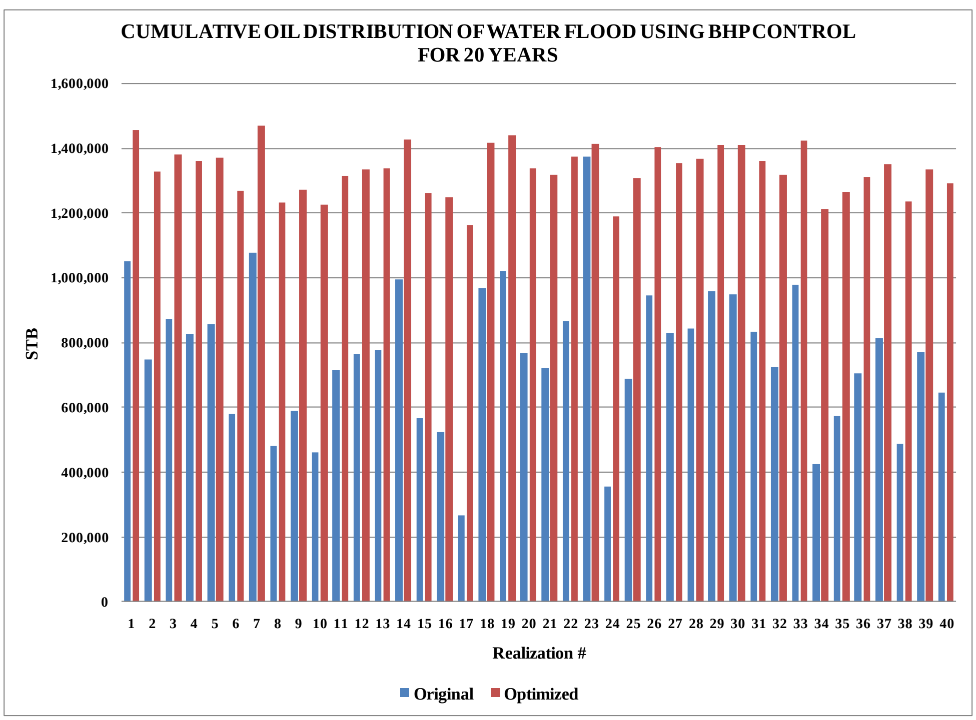


Figure 46: Cumulative Oil Distribution of Water Flood for 20 Years

From the preceding graphs, it can be seen that the optimization successfully improved the NPV and the cumulative oil produced for each ensemble member. Although the optimization can successfully improve the NPV, it is important to examine the field pressure and oil saturation to see if loosely constrained controls hinder good reservoir management. The following figures illustrate the field oil saturation (Figure 47) and aerial view (Figure 48) for the 1st ensemble member of the optimized water flood.

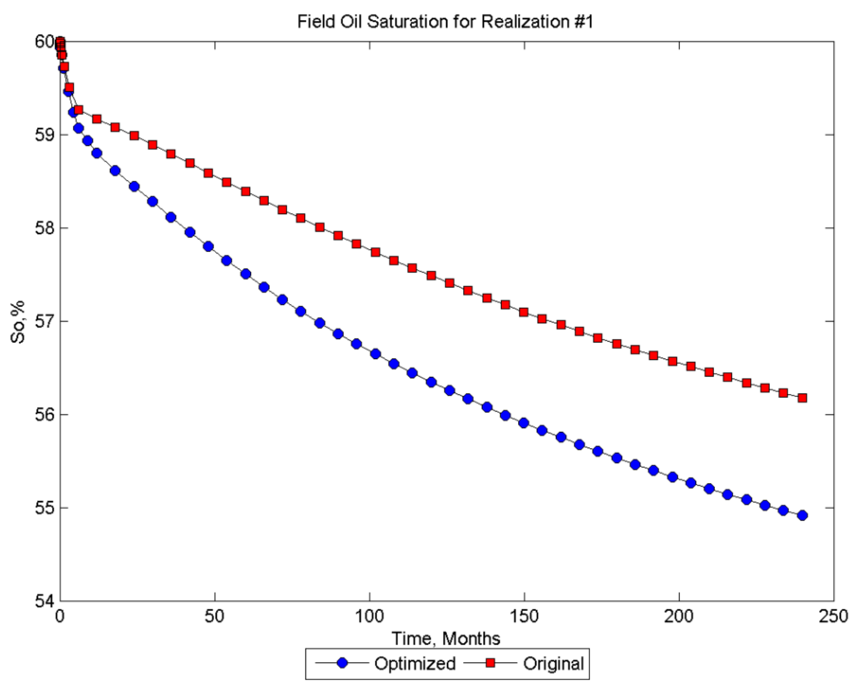


Figure 47: Field Oil Saturation for Water Flood for 20 Years (1st Realization)

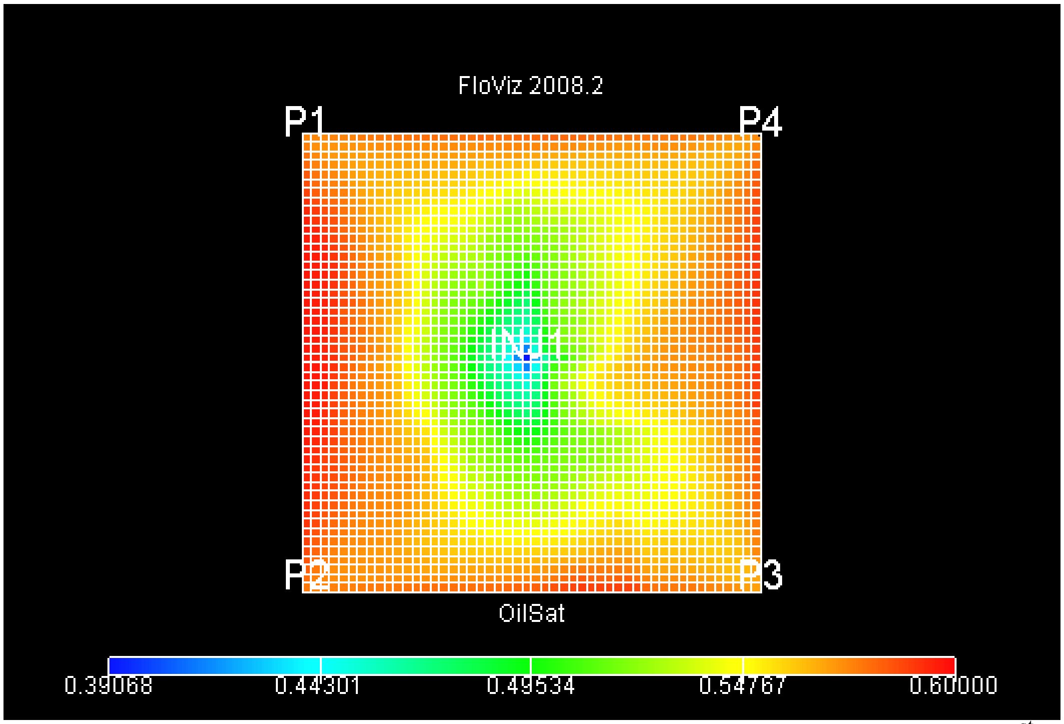
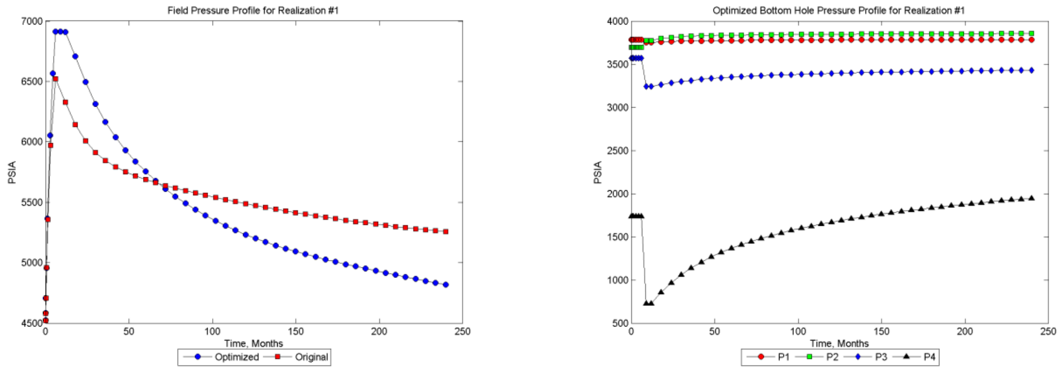


Figure 48: Field Oil Distribution for Optimized Water Flood for 20 Years (1st Realization)

The field oil saturation plots reveal that the oil saturation did not reduce substantially for the optimized case. The field pressure, however, shows the detrimental effects of loosely defined controls. These effects are illustrated in Figure 49.



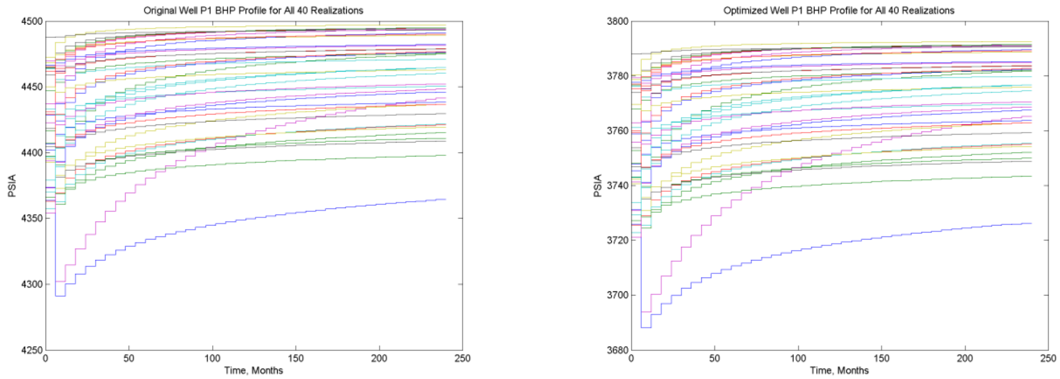
(a) (b)

Figure 49: Field and Bottomhole Pressures for Optimized Water Flood for 20 Years (1st Realization): (a) Field Pressure, (b) Bottomhole Pressure for All Production Wells

The observed high reservoir pressure is a result of high injection rates combined with bottomhole pressures that do not allow enough liquid to produce. Pressure buildup occurs because the injection rate is too high and the production rate is too low. To alleviate this effect of pressure buildup it would of have been wiser to limit the maximum control of the producer to values much less than 4500 psia.

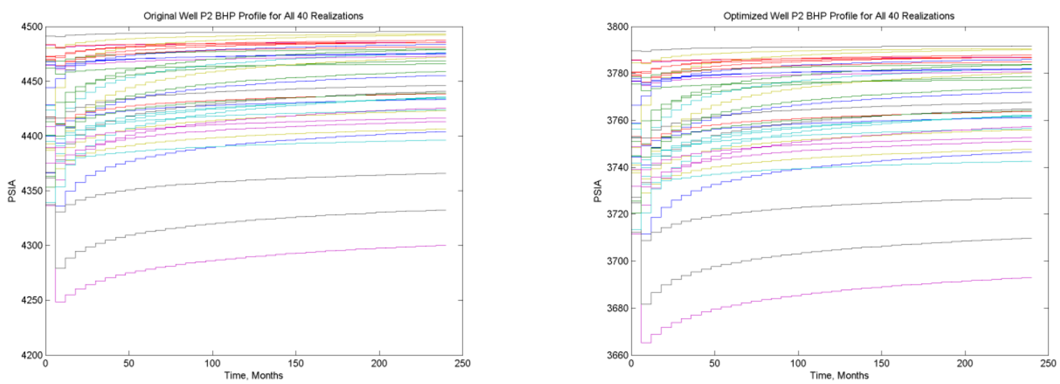
12.1.2 SURFACTANT FLOOD FOR 20 YEARS WITH BOTTOMHOLE PRESSURE CONTROL OF THE PRODUCER

Applying the defined DCA parameters in Table 13 and constraints for the production time of 20 years in Table 14; Figure 50, Figure 51, Figure 52, Figure 53, and Figure 54 illustrate the original and optimized controls for the surfactant flood for all ensemble members.



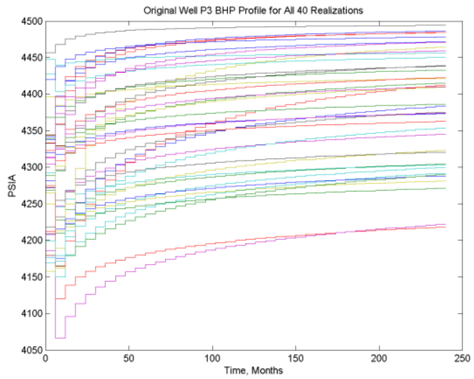
(a) (b)

Figure 50: Well P1 Controls for Surfactant Flood for 20 Years: (a) Original, (b) Optimized

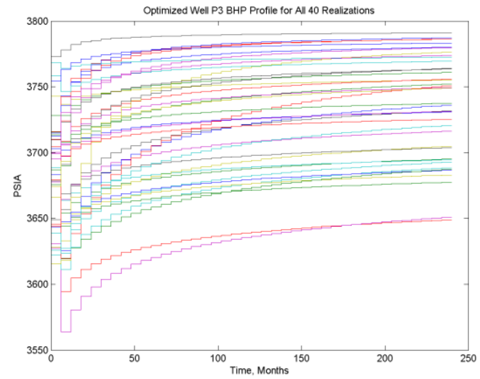


(a) (b)

Figure 51: Well P2 Controls for Surfactant Flood for 20 Years: (a) Original, (b) Optimized

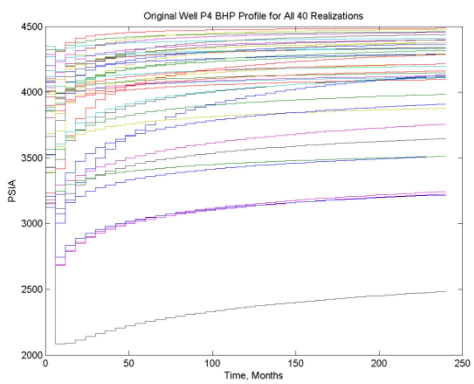


(a)

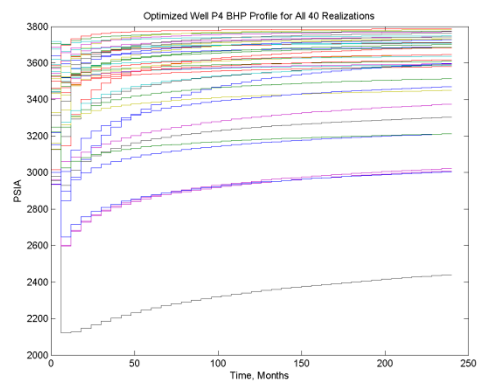


(b)

Figure 52: Well P3 Controls for Surfactant Flood for 20 Years: (a) Original, (b) Optimized

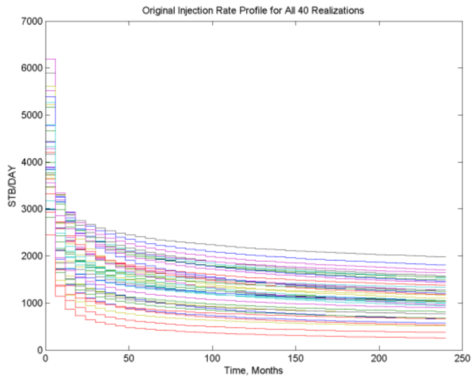


(a)

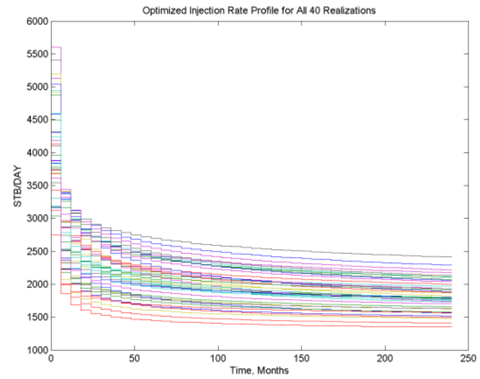


(b)

Figure 53: Well P4 Controls for Surfactant Flood for 20 Years: (a) Original, (b) Optimized



(a)



(b)

Figure 54: Injection Rate Controls for Surfactamt Flood for 20 Years: (a) Original, (b) Optimized

It can be seen that the trends in the original controls are maintained in the optimized controls. The optimal surfactant concentration for all the ensemble member of controls was 1.77 lb/STB.

Figure 55 and Figure 56 illustrate the resultant NPV and cumulative oil distribution obtained from optimizing the process for all ensemble members.

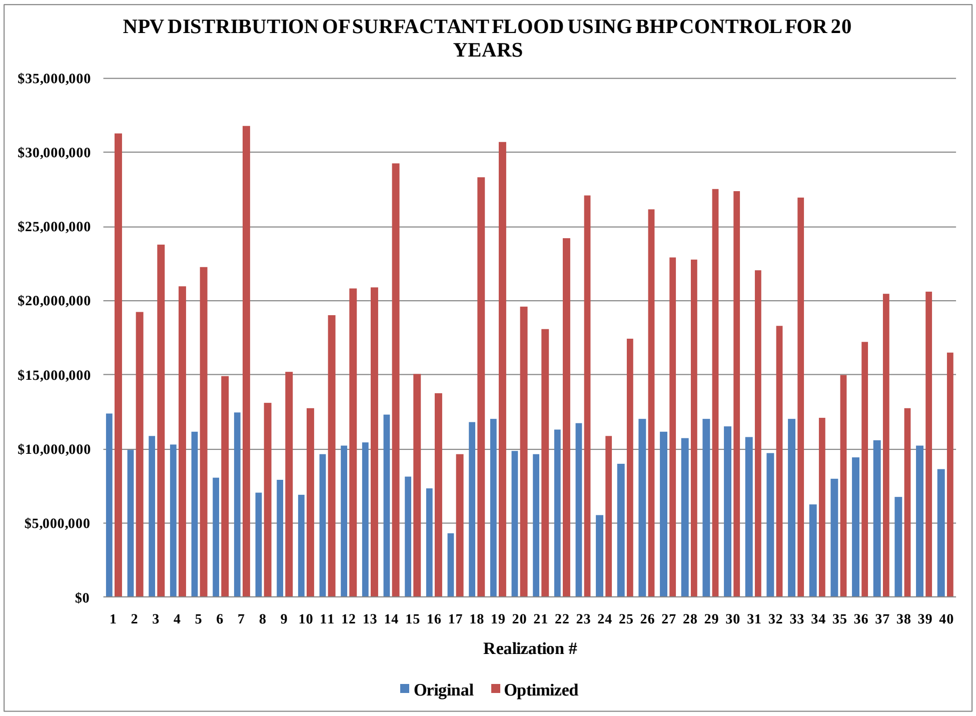


Figure 55: NPV Distribution of Surfactant Flood for 20 Years

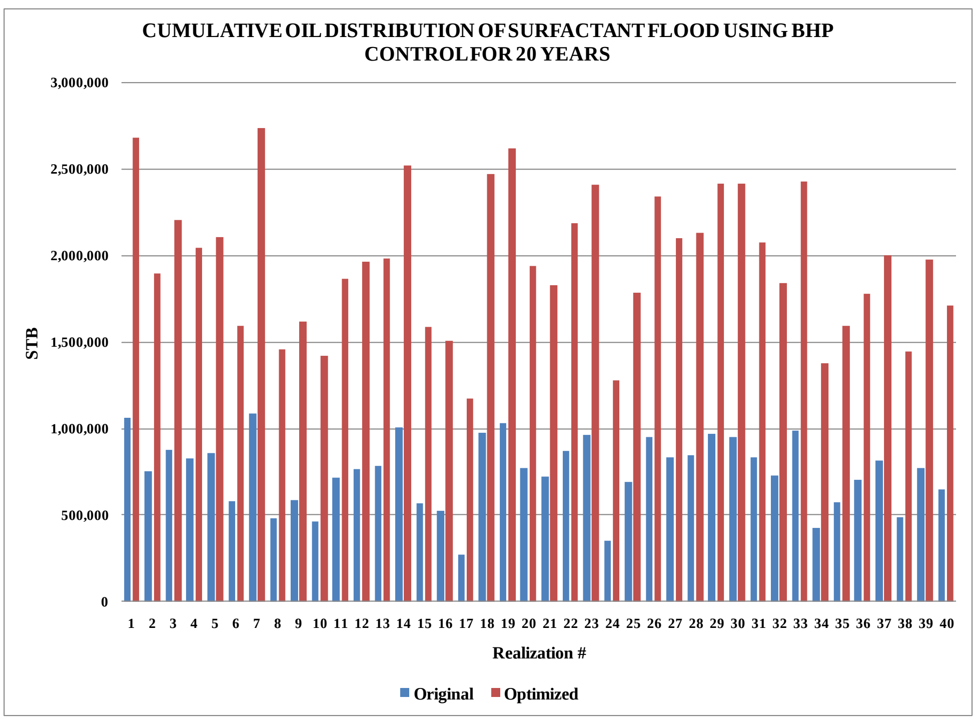


Figure 56: Cumulative Oil Distribution of Surfactant Flood for 20 Years

The preceding graphs show that the optimization was successful in improving the NPV and the cumulative oil produced for each ensemble member. Although the optimization is successful in improving the NPV, it is important to look at the field pressure and oil saturation to observe the effects of loosely constrained controls. The following figures are the average oil saturation (Figure 57) and aerial view (Figure 58) for the first ensemble member of the optimized surfactant flood.

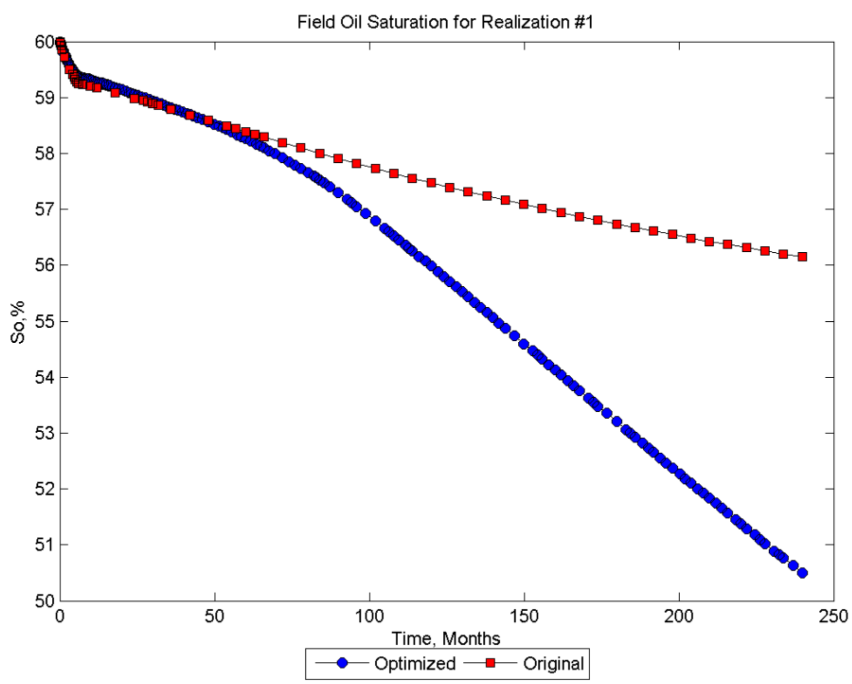


Figure 57: Field Oil Saturation for Surfactant Flood for 20 Years (1st Realization)

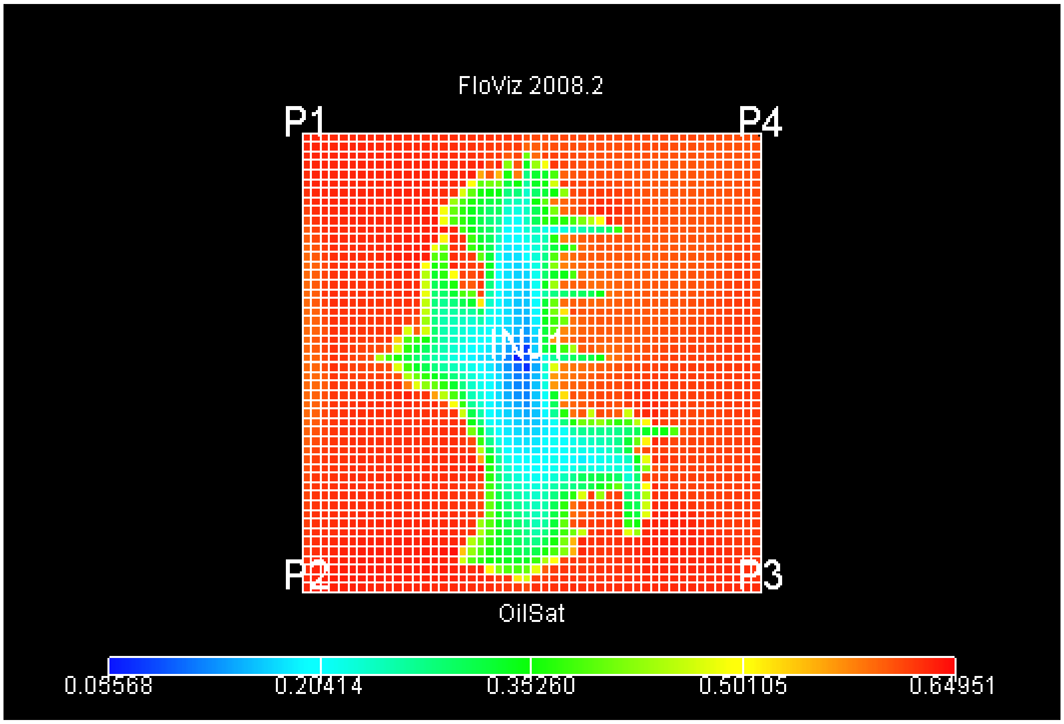
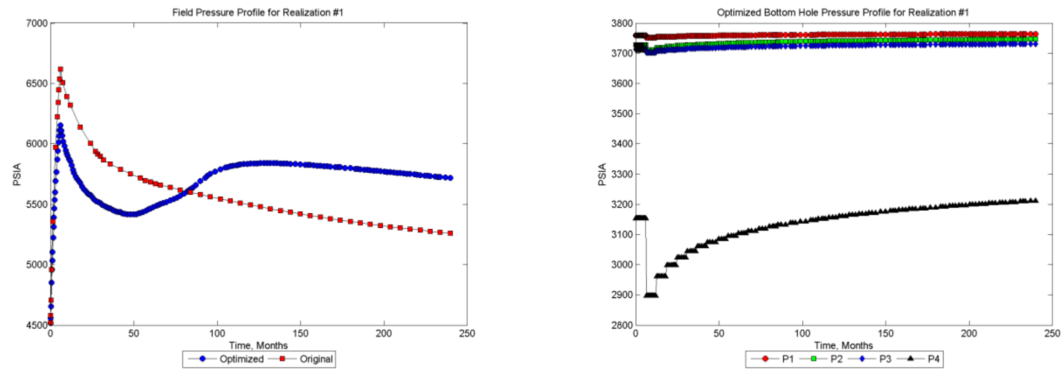


Figure 58: Field Oil Distribution for Optimized Surfactant Flood for 20 Years (1st Realization)

The field oil saturation distribution plot illustrates that the oil saturation reduces past the water flood residual for areas corresponding to the permeability distribution see (see Figure 14). The field oil saturation, however, illustrates that the oil saturation reduces enough to sustain the project through its life (i.e. NPV is positive and therefore the project can sustain itself).

The field pressure shows the detrimental effects of loosely defined controls. These effects can be seen in Figure 59 along with the optimized bottomhole pressure of each producer.



(a) (b)

Figure 59: Field and Bottomhole Pressures for Optimized Surfactant Flood for 20 Years (1st Realization): (a) Field Pressure, (b) Bottomhole Pressure for All Production Wells

High reservoir pressures observed in the previous figure are a result of high injection rates combined with bottomhole pressures that do not allow enough liquid to produce. Pressure buildup occurs because the injection rate is too high and the production rate is too low. To mollify this effect of pressure buildup the maximum control of the producer needs to be limited to values much less than 4500 psia.

12.1.3 COMPARISON BETWEEN WATER FLOOD AND SURFACTANT FLOOD FOR 20 YEARS OF PRODUCTION

Loosely defined control constraints may result in detrimental effects to the reservoir such as too high a reservoir pressure, verifying the importance of choosing constraints consistent with reservoir limits. Regardless of the effects of using the constraints for a time scale of 20 years, the surfactant flood is more economic than the water flood. Proof of this observation is verified by Table 15. (Note BHP_SURF_20yrs corresponds to a 20 year surfactant flood using bottomhole pressure as the producer control while BHP_WATER_20yrs corresponds to a 20 year water flood using bottomhole pressure as the producer control).

Table 15: NPV Summary for Production Time of 20 Years

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized	Increase in 1st Realization	Increase in Average
BHP_SURF_20yrs	$12,401,997	$9,763,451	$31,298,509	$20,473,815	152.37%	109.70%
BHP_WATER_20yrs	$14,814,147	$11,454,504	$17,598,149	$18,011,221	18.79%	57.24%

For the first ensemble member the cumulative distribution curve for both optimized methods for is illustrated in Figure 60 and Figure 61 (with the mode economic parameter and mean permeability field indicated on the plot).

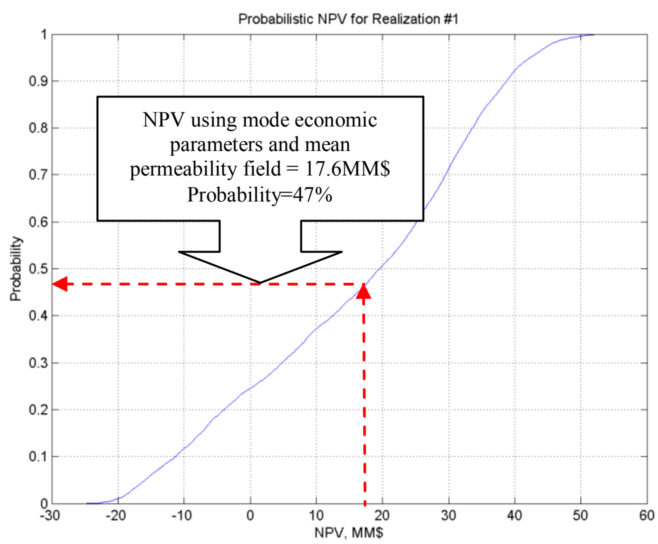


Figure 60: Cumulative Distribution Curve for NPV for Water Flood for Production Time of 20 Years

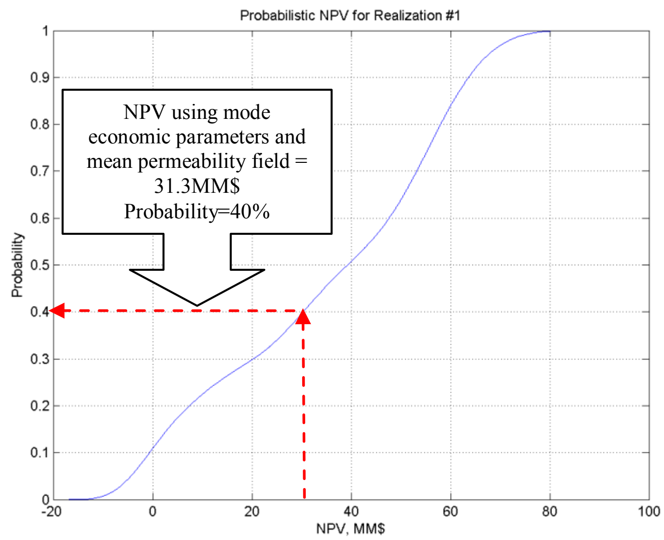


Figure 61: Cumulative Distribution Curve for NPV for Surfactant Flood for Production Time of 20 Years

NPV is an indicator of how well the EOR process can add value. The cumulative distribution curve is a ranking of the possible scenarios incorporating permeability and economics. The cumulative distribution curve generated as a result of the sampling of permeability fields and economic parameters is an indication of the possible NPVs that can occur if a set of permeability field and economic parameters were implemented at the start of the project. This means that for the surfactant flood 40% of the scenarios incorporating permeability and economics are worse than using the mode economic parameters and mean permeability field. Similarly, 47% of the scenarios incorporating permeability and economics for the water flood are worse than using the mode economic parameters and mean permeability field. Although there are regions in the cumulative distribution curve that are negative, these regions correspond to less favorable economics such as low oil price, expensive surfactant, expensive water injection cost, and expensive water treatment cost. These negatives NPVs are not an indication of the failure of the project because using the mode economic parameters and mean permeability field the NPV is positive. These negative NPVs represent a warning to have favorable economics to ensure project success. From a financial viewpoint, the surfactant flood is more economic than the water flood for a production period of 20 years.

12.2 PRODUCTION TIME OF 40 YEARS WITH LIQUID RATE CONTROL OF THE PRODUCER

Production time of 40 years with bottomhole pressure control of the producer was analyzed with tightly defined constraints. Tightly defined constraints correspond to the situation in which the controls for the producer and injector are restricted to have a narrow range of values. The purpose of studying tightly defined controls is to convey that controls have to be well defined to ensure reasonable reservoir management during optimization. The following tables are the DCA parameters (Table 16) and the constraints (Table 17) for this scenario.

Table 16: DCA Parameters for Production Time of 40 Years

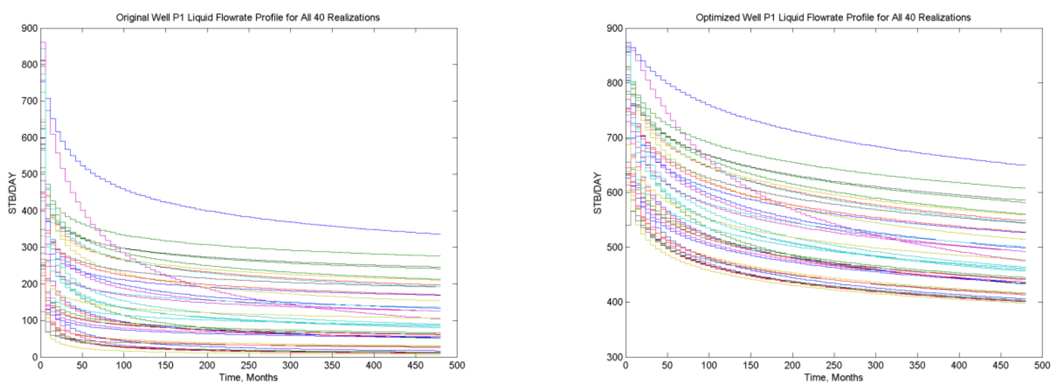
	mean	minimum	max
Q_{init}, STB/DAY	500	100	900
b	5	1	9
D_{init}, /year	5	0.1	9.9

Table 17: Constraints for Production Time of 40 Years

	Minimum	Maximum	Mean	Standard Deviation
Liquid Rate, STB/DAY	10	500	255	245
Injection Rate, STB/DAY	40	2000	1020	980
Surfactant Concentration, LB/STB	0.01	10	5.005	4.995

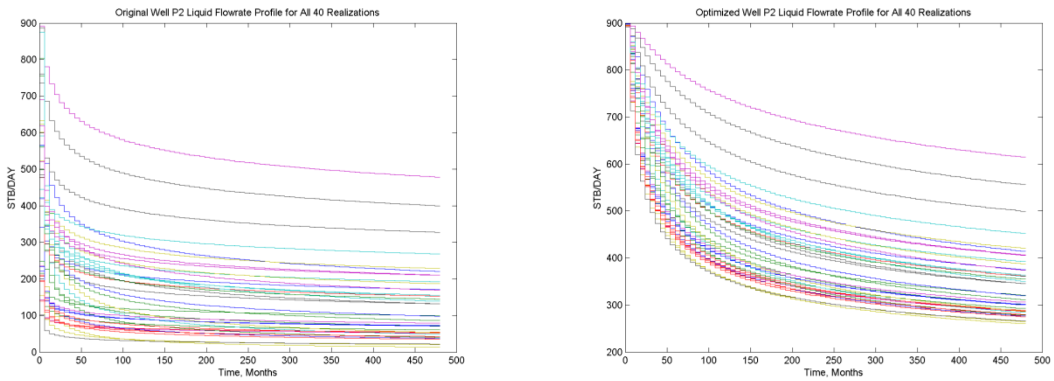
12.2.1 WATER FLOOD FOR 40 YEARS WITH LIQUID RATE CONTROL OF THE PRODUCER

Using the defined original DCA parameters and constraints for the production time of 40 years; Figure 62, Figure 63, Figure 64, Figure 65, and Figure 66 illustrate the original and optimized controls for the water flood.



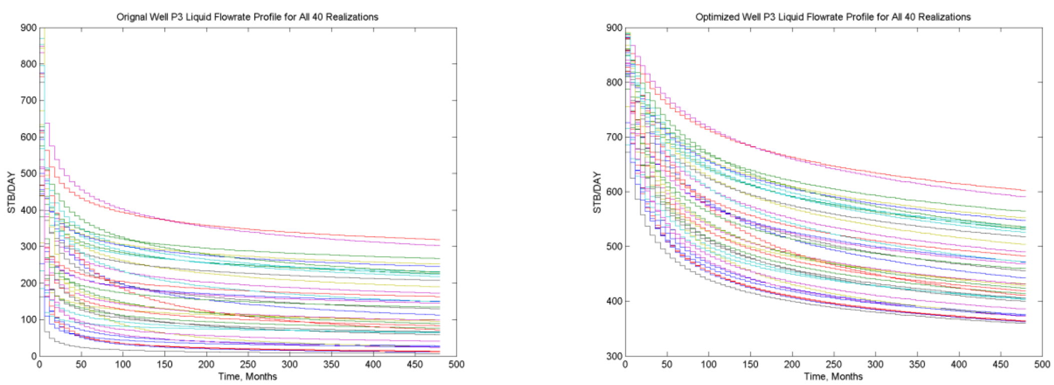
(a) (b)

Figure 62: Well P1 Controls for Water Flood for 40 Years: (a) Original, (b) Optimized



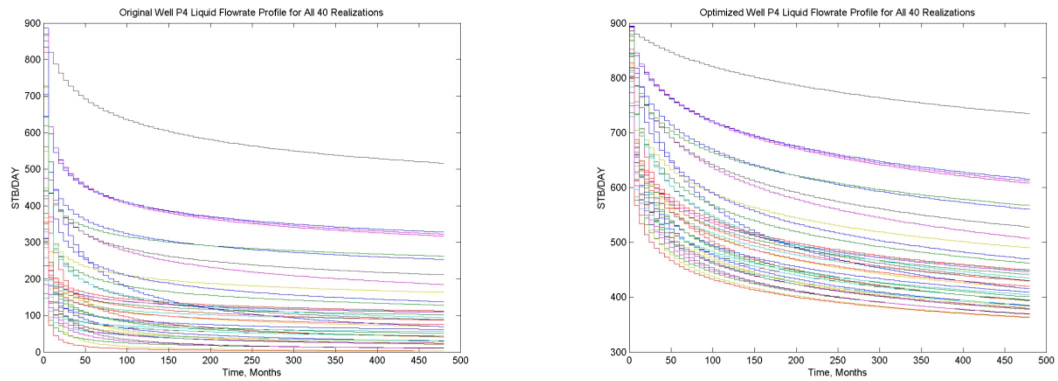
(a) (b)

Figure 63: Well P2 Controls for Water Flood for 40 Years: (a) Original, (b) Optimized



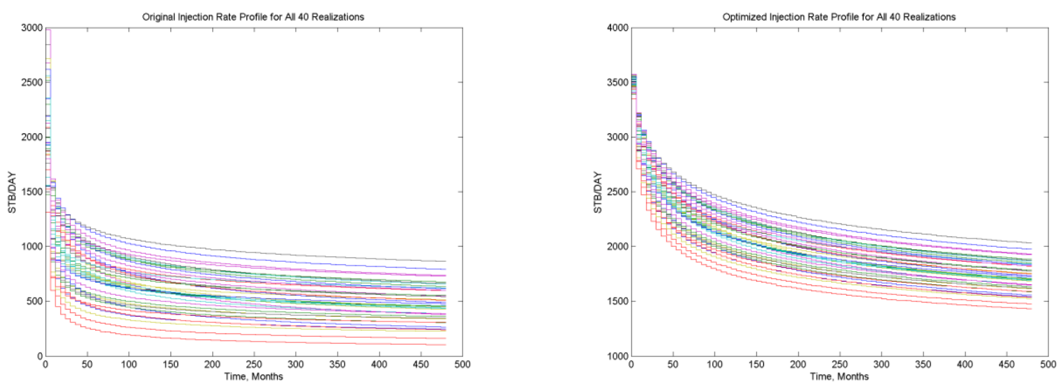
(a) (b)

Figure 64: Well P3 Controls for Water Flood for 40 Years: (a) Original, (b) Optimized



(a) (b)

Figure 65: Well P4 Controls for Water Flood for 40 Years: (a) Original, (b) Optimized



(a) (b)

Figure 66: Injection Rate Controls for Water Flood for 40 Years: (a) Original, (b) Optimized

It is apparent that the trends in the original controls are maintained in the optimized controls. Another observation is that the variability among the realization of the controls is reduced.

Figure 67 and Figure 68 illustrate the resultant NPV and cumulative oil distribution obtained from the optimization of all ensemble members.

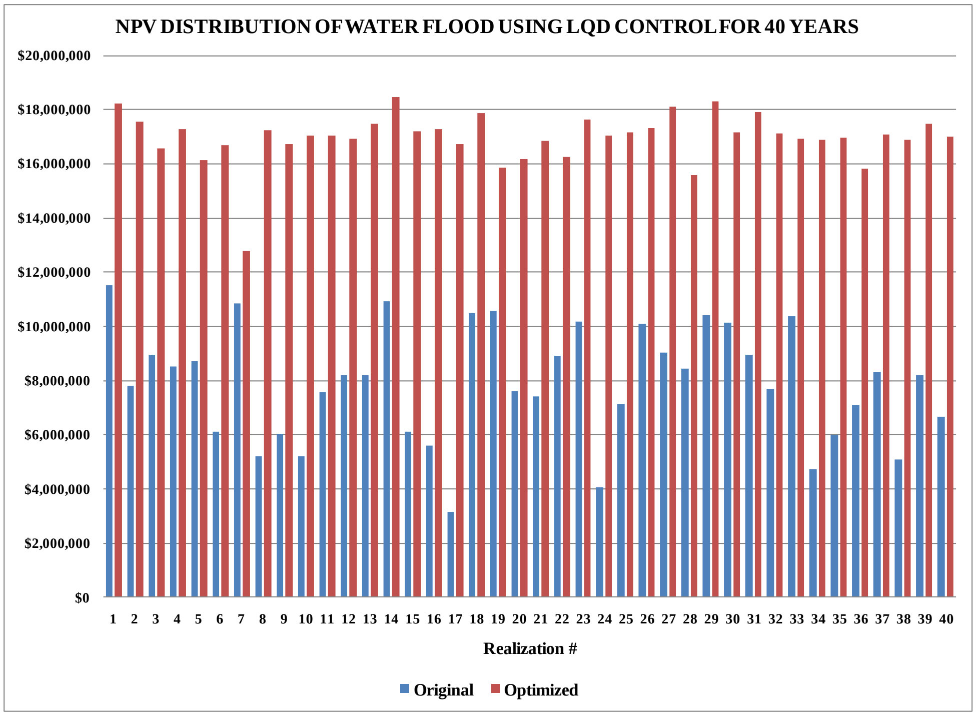


Figure 67: NPV Distribution of Water Flood for 40 Years

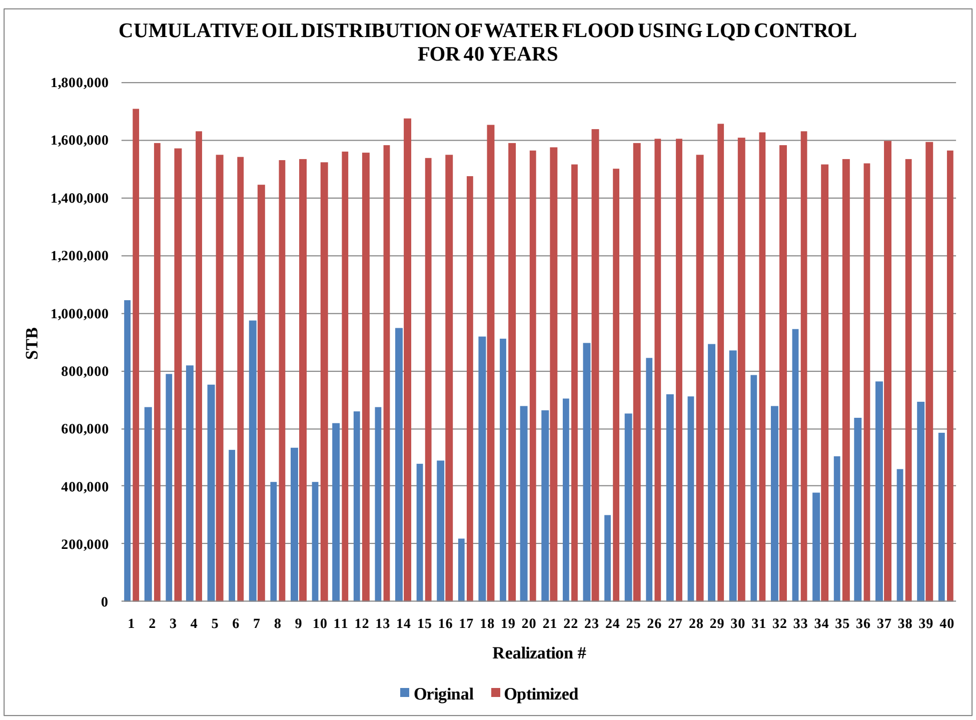


Figure 68: Cumulative Oil Distribution of Water Flood for 40 Years

The preceding graphs show that the optimization was successful in improving the NPV and the cumulative oil for each ensemble member. Optimization was successful in improving the NPV, but it is important to investigate the field pressure and oil saturation to understand the consequences of tightly constrained controls. The following figures are the average oil saturation (Figure 69) and aerial view (Figure 70) for the first ensemble member of the optimized 40 year water flood.

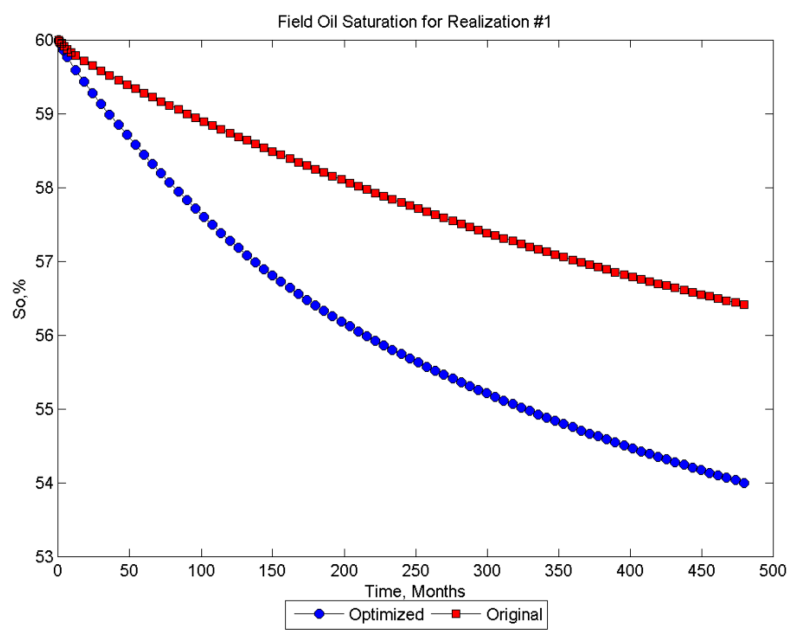


Figure 69: Field Oil Saturation for Water Flood for 40 Years (1st Realization)

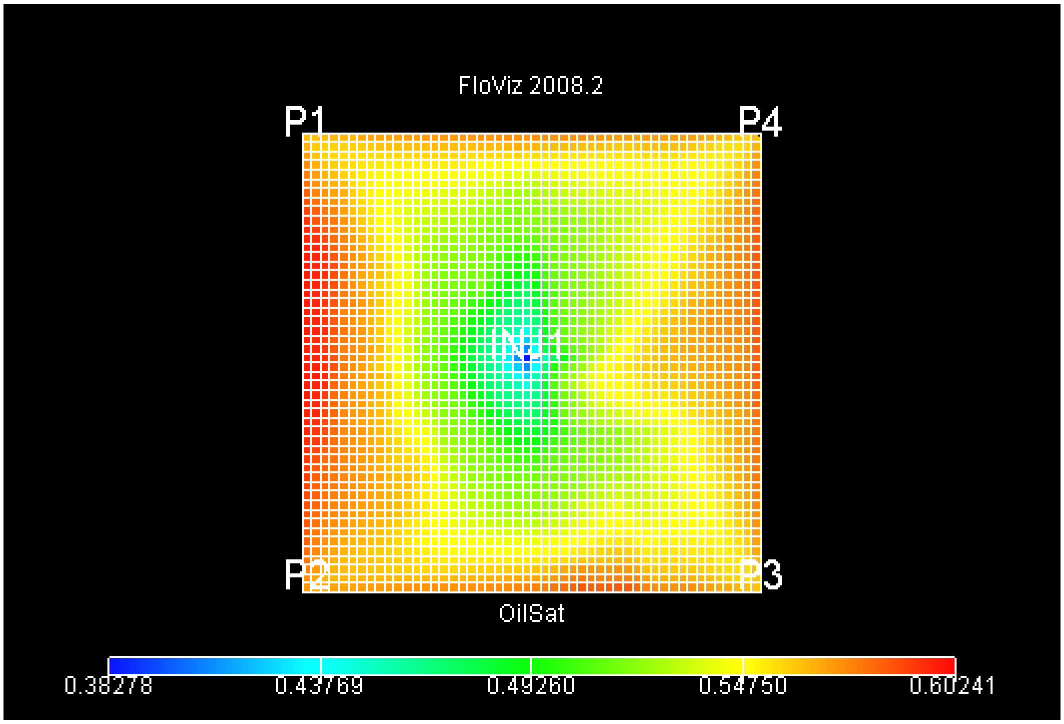


Figure 70: Field Oil Distribution for Optimized Water Flood for 40 Years (1st Realization)

The oil saturation reduces enough to sustain the project through its life (i.e. NPV is positive as a result of profit from increased oil production). In addition to the oil saturation decreasing to maintain favorable economics, the field pressure shows the effects of tightly defined controls. These effects are illustrated in the following figure (Figure 71) along with the optimized liquid rate of each producer (Figure 72).

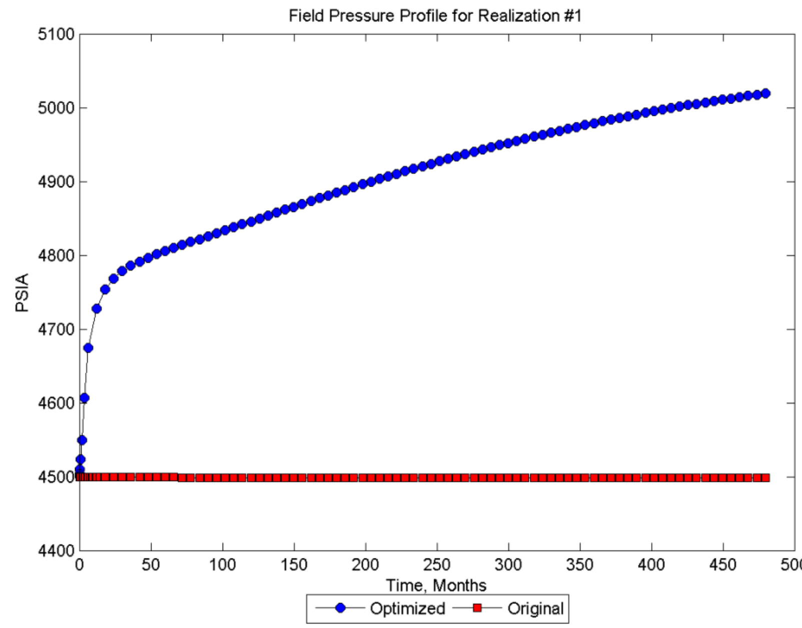
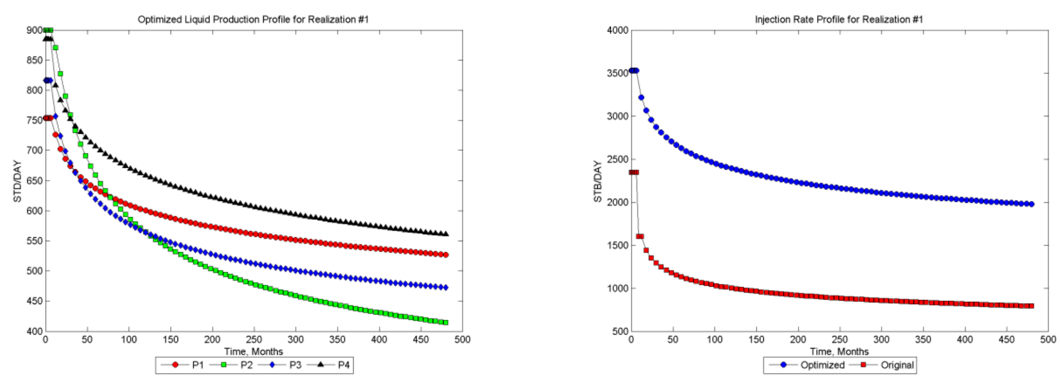


Figure 71: Field Pressure for Water Flood for 40 Years (1st Realization)



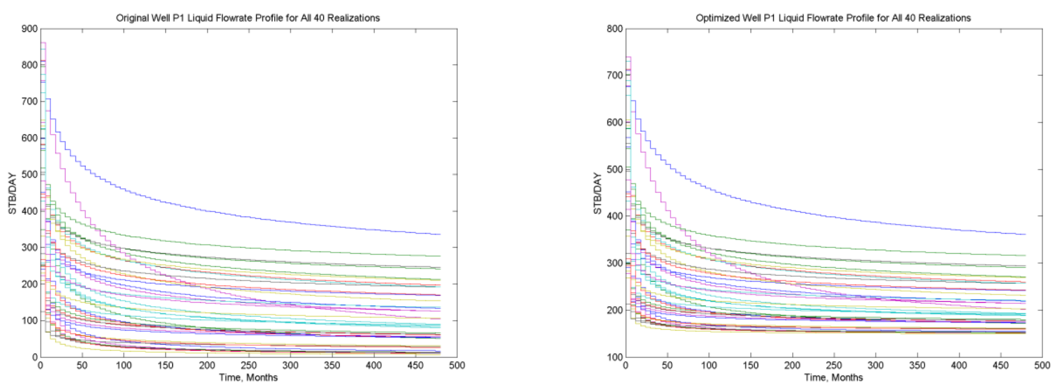
(a) (b)

Figure 72: Optimized Liquid Production and Injection Rate for Water Flood for 40 Years (1st Realization): (a) Optimized Liquid Production, (b) Injection Rate

The pressure increase observed in the optimized case is a result of a higher injection rate than the total production rate. Tightly constrained controls for the water flood show favorable reservoir management because the reservoir pressure increase in the optimized case is much less than in the case of loosely constrained controls (see Figure 49).

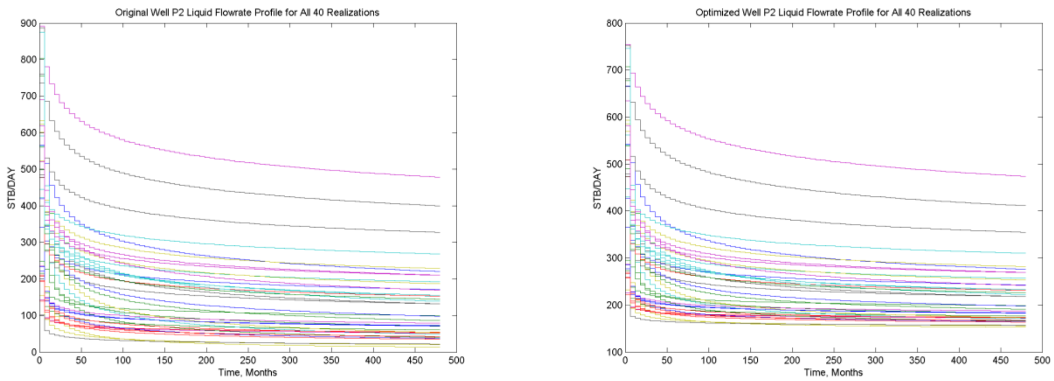
12.2.2 SURFACTANT FLOOD FOR 40 YEARS WITH LIQUID RATE CONTROL OF THE PROUDCER

Applying the defined original DCA parameters and constraints for the production time of 40 years; Figure 73, Figure 74, Figure 75, Figure 76, and Figure 77 illustrate the original and optimized controls for the surfactant flood.



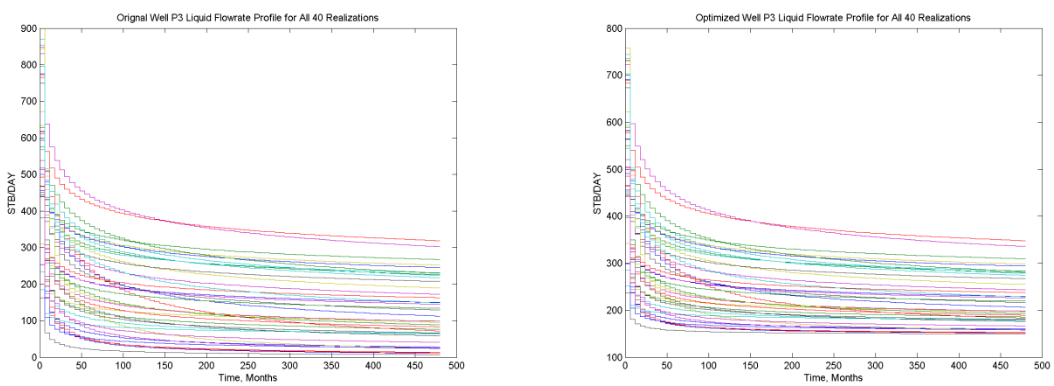
(a) (b)

Figure 73: Well P1 Controls for Surfactant Flood for 40 Years: (a) Original, (b) Optimized



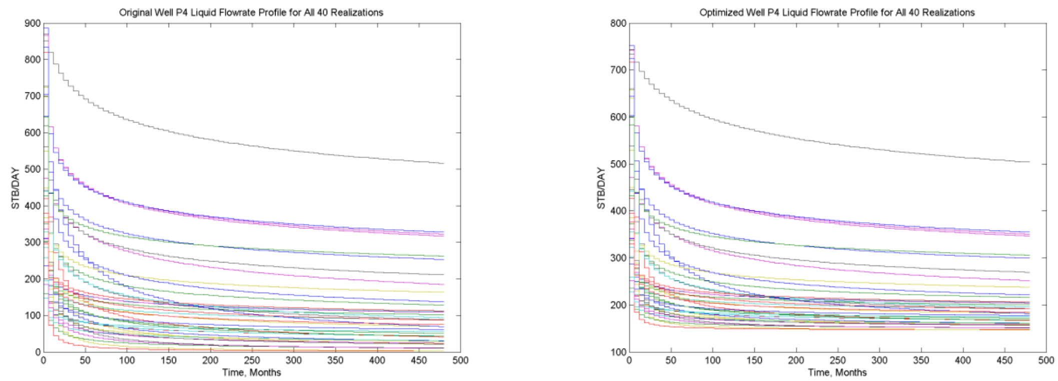
(a) (b)

Figure 74: Well P2 Controls for Surfactant Flood for 40 Years: (a) Original, (b) Optimized



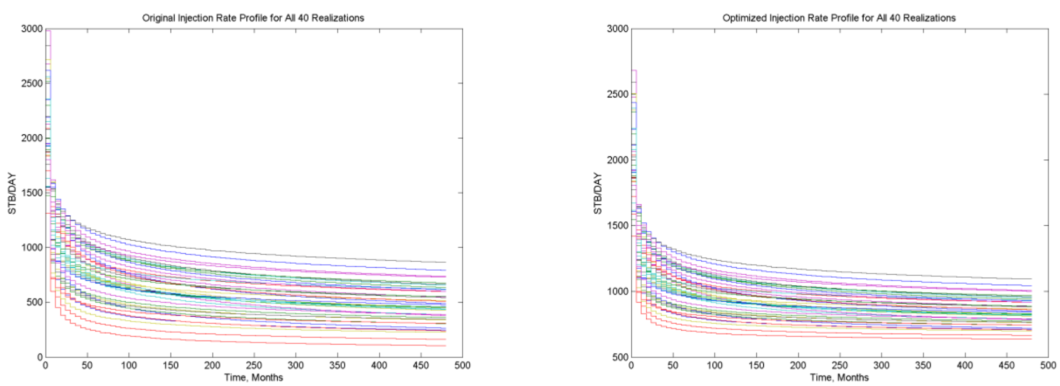
(a) (b)

Figure 75: Well P3 Controls for Surfactant Flood for 40 Years: (a) Original, (b) Optimized



(a) (b)

Figure 76: Well P4 Controls for Surfactant Flood for 40 Years: (a) Original, (b) Optimized



(a) (b)

Figure 77: Injection Rate Controls for Surfactant Flood for 40 Years: (a) Original, (b) Optimized

The previous figures illustrate that the trends in the original controls are maintained in the optimized controls. The optimal surfactant concentration for all the ensemble member of controls was 1.77 lb/STB. The optimal concentration of the surfactant for 40 years of production case is the same concentration for the 20 years of production case. This phenomenon is due to the simple surfactant properties used in this research. Many of the properties were linear with respect to concentration (i.e. the interfacial tension). Although the concentration for these two cases is identical, the mass rate of surfactant injected is not identical because the optimized ensemble of injection rates for both cases is different.

Figure 78 and Figure 79 illustrate the resultant NPV and cumulative oil distribution obtained from the optimization process for all ensemble members.

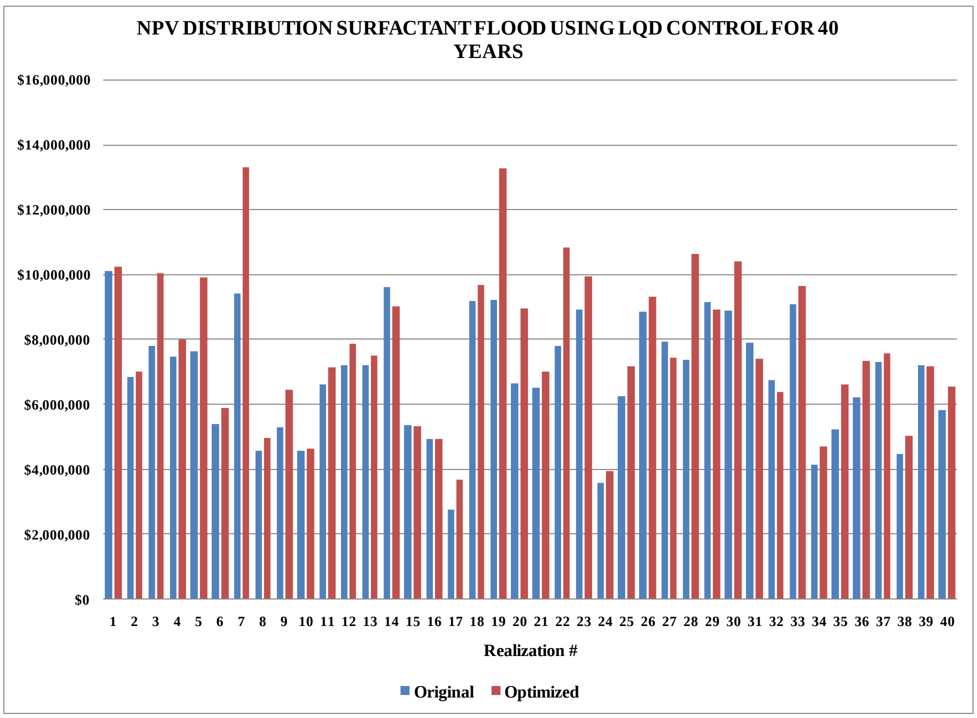


Figure 78: NPV Distribution of Surfactant Flood for 40 Years

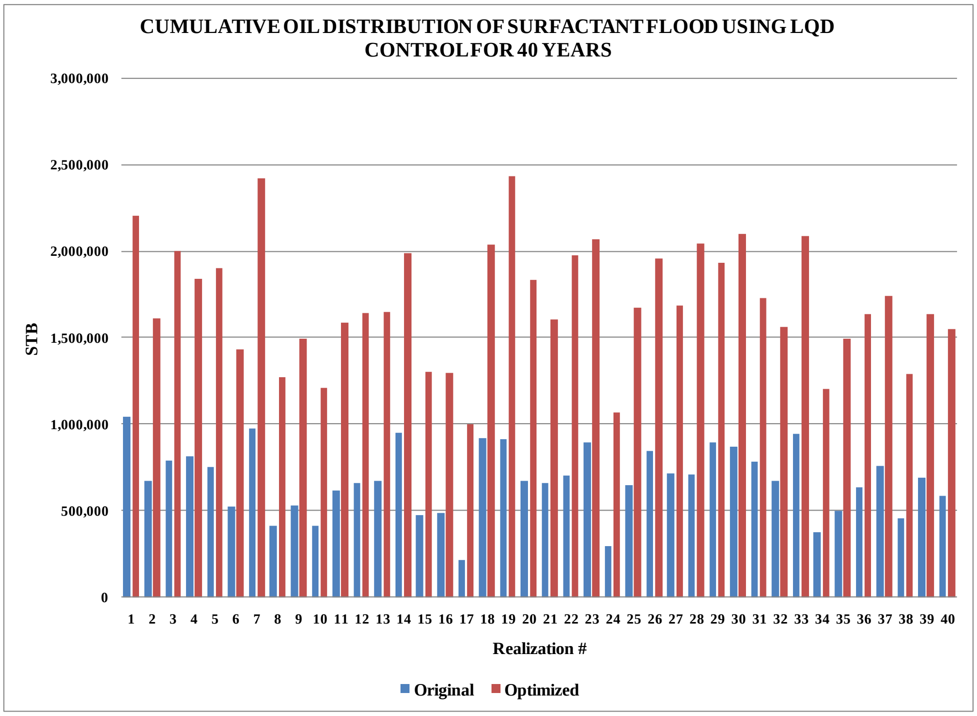
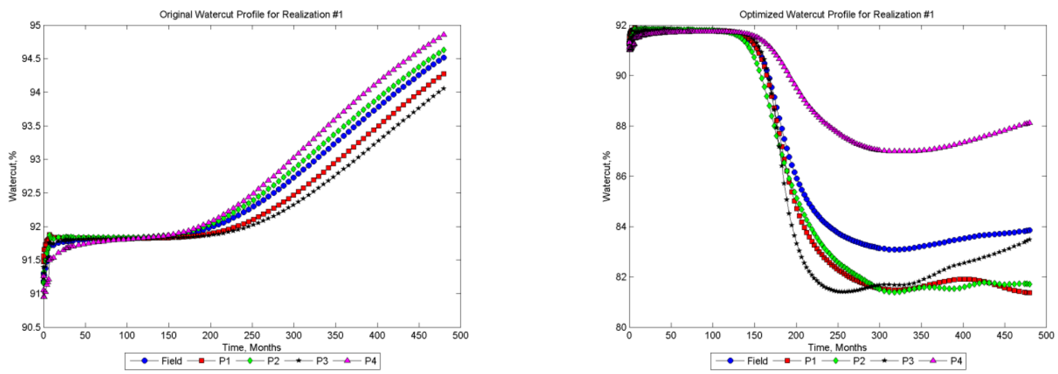


Figure 79: Cumulative Oil Distribution of Surfactant Flood for 40 Years

The preceding graphs show that the optimization was successful in improving the average NPV and the average cumulative oil produced for each ensemble member. Another observation conveyed in the plot of the NPV and cumulative oil is that while there is a substantial increase in the cumulative oil for all the ensemble members the increase in the NPV is not too significant for all the ensemble members. These observations show that increased production in oil does not guarantee equal increase in NPV (for example see realization 1 in Figure 78 and Figure 79). This implies that increased production of oil does not correlate in increasing the value of the process. Factors such as water handling cost, surfactant cost, and water treatment cost can adversely affect the positive impact of increased oil production.

The optimization improved the reservoir management of the produced water. Improved water handling is illustrated in the comparison between original and optimized watercut (Figure 80).



(a) (b)

Figure 80: Watercut for Surfactant Flood for 40 Years (1st Realization): (a) Original, (b) Optimized

It is apparent that optimization reduced the amount of water produced for the life of the project. This can be interpreted as the optimization method reducing the volume of water and hence treatment cost so that NPV can remain positive for the project.

Optimization also improved the reservoir management of the field oil saturation. Field saturation for the original and optimized surfactant flood can be seen in Figure 81 followed by the optimized field oil saturation distribution after 40 years (Figure 82).

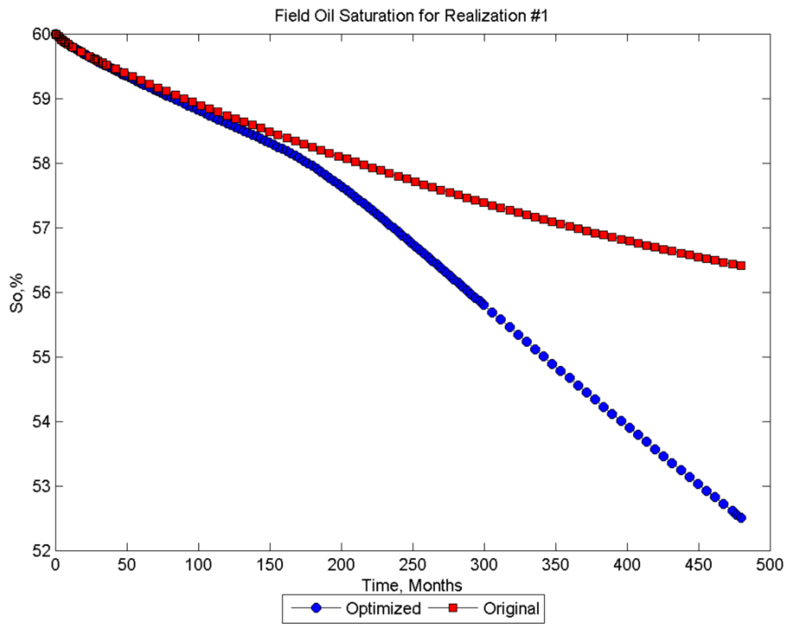


Figure 81: Field Oil Saturation for Surfactant Flood for 40 Years (1st Realization)

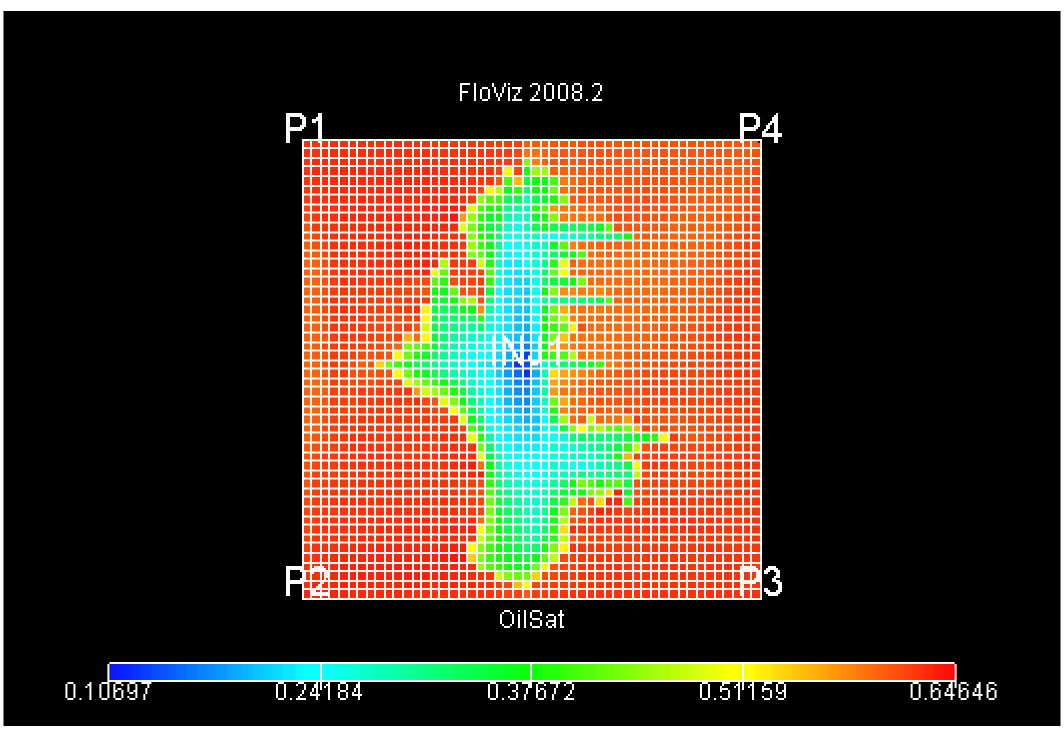


Figure 82: Field Oil Distribution for Optimized Surfactant Flood for 40 Years (1st Realization)

From both field oil saturation plots, it can be seen that the field produces past the water flood residual oil saturation of 30% for the optimized case for select areas corresponding to the permeability distribution in the field (see Figure 14). Overall field saturation, however, does not produce past the residual oil saturation. This implies that the NPV after 40 years is not necessary maximized when the final field saturation is below the water flood residual oil saturation. When trying to produce past the water flood residual oil saturation, water cut may increase and thus increase the cost of the water treatment. This water treatment cost prevents the optimization from producing past the water flood residual oil saturation. The main conclusion to draw from observing the field saturation is that increased cumulative oil does not equate to improved economics.

The field pressure shows the advantages of tightly defined constraints. These effects can be seen in Figure 83 along with the optimized liquid rate of each producer and the optimized injection rate (Figure 84).

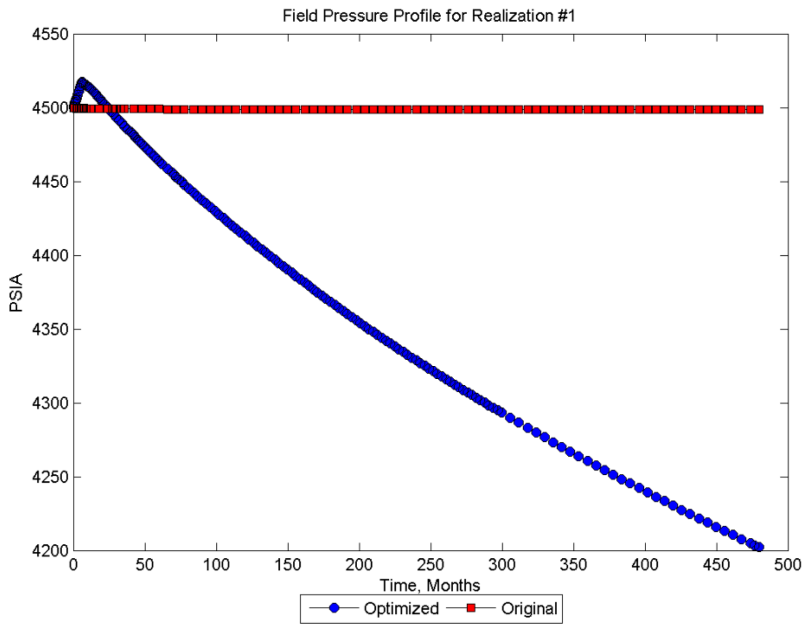
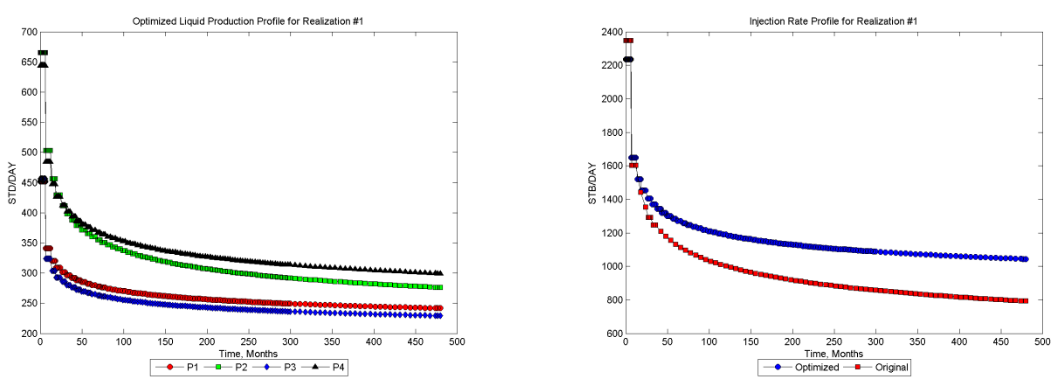


Figure 83: Field Pressure for Surfactant Flood for 40 Years (1st Realization)



(a) (b)

Figure 84: Optimized Liquid Production and Injection Rate for Surfactant Flood for 40 Years (1st Realization): (a) Optimized Liquid Production, (b) Injection Rate

It is apparent that the optimization with tightly defined controls maintained a proper balance between injection and production for the 40 year period. This is observed in the manageable field pressure through time for the optimized case.

12.2.3 COMPARISON BETWEEN WATER FLOOD AND SURFACTANT FLOOD FOR 40 YEARS OF PRODUCTION

Tightly defined control constraints resulted in good reservoir management while having a positive NPV at the end of the project. Both the water flood and surfactant flood were successful at the end of the 40 years, but the water flood adds more economic value than the surfactant flood. This observation is conveyed in Table 18. (Note LQD_SURF_40yrs corresponds to a 40 year surfactant flood using liquid production rate as the producer control while LQD_WATER_40yrs corresponds to a 40 year water flood using liquid production rate as the producer control).

Table 18: NPV Summary for Production Time of 40 Years

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized	Increase in 1st Realization	Increase in Average
LQD_SURF_40yrs	$10,135,908	$6,949,871	$10,266,029	$7,814,655	1.28%	12.44%
LQD_WATER_40yrs	$11,531,566	$7,906,018	$18,232,749	$16,970,953	58.11%	114.66%

The water flood shows to have an advantage over the surfactant flood for 40 years of production when NPV is used as an indicator of success but the cumulative oil for both methods shows a production increases after optimization of NPV. This is seen in Table 19.

Table 19: Cumulative Oil Summary for Production Time of 40 Years

Case	1st Realization, Original, STB	Average, Original, STB	1st Realization, Optimized, STB	Average, Optimized, STB	Increase in 1st Realization	Increase in Average
LQD_SURF_40yrs	1,047,383	681,510	2,206,360	1,707,843	110.65%	150.60%
LQD_WATER_40yrs	1,045,627	680,587	1,707,625	1,573,665	63.31%	131.22%

This illustrates that cumulative oil increase does not mean that there will be an equal increase in NPV. The cumulative oil shows that the surfactant flood produces more oil than the water flood, but the NPV shows that the surfactant flood cost is much higher than the water flood. The NPV, however, shows that the water flood is more profitable than the surfactant flood. The cumulative distribution curve of the NPV for both optimized methods for the 1st ensemble member can be seen in Figure 85 and Figure 86.

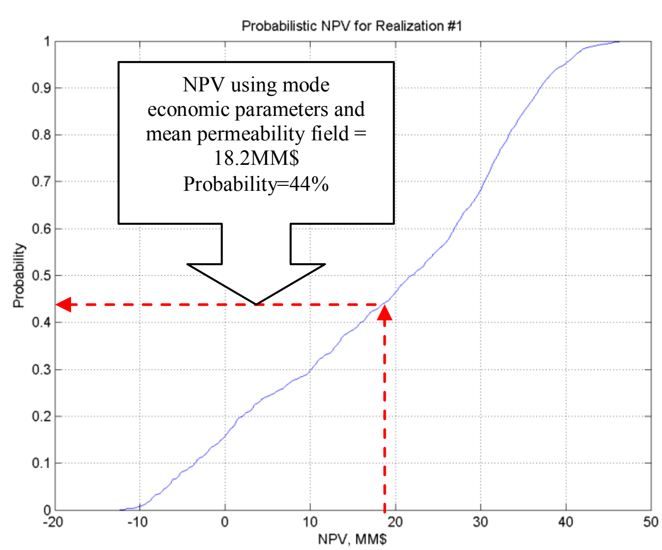


Figure 85: Cumulative Distribution Curve for NPV for Water Flood for Production Time of 40 Years

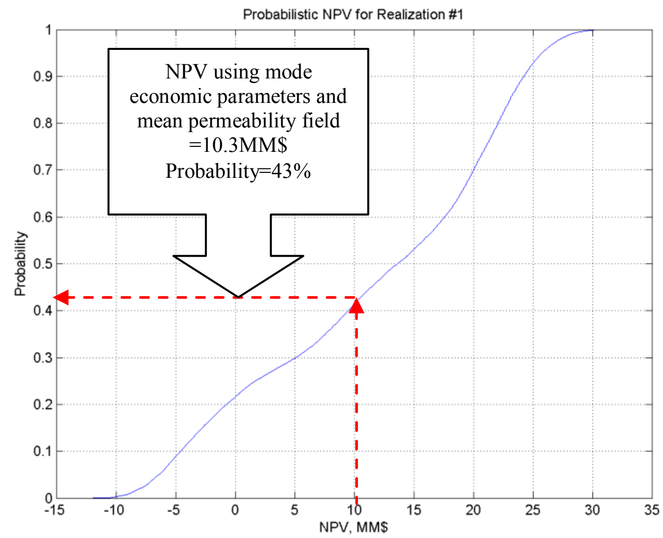


Figure 86: Cumulative Distribution Curve for NPV for Surfactant Flood for Production Time of 40 Years

For the surfactant flood, 43% of the scenarios incorporating permeability and economics are worse than using the mode economic parameters and mean permeability field. Similarly, 44% of the scenarios incorporating permeability and economics for the water flood are worse than using the mode economic parameters and mean permeability field. There are regions in the cumulative distribution curve that are negative, these regions correspond to less favorable economics such as low oil price, expensive surfactant, expensive water injection cost, and expensive water treatment cost. These negatives NPVs are not an indication of the failure of the project because using the mode economic parameters and mean permeability field the NPV is positive. These negative NPVs represent a warning to have favorable economics to ensure project success. From an economic viewpoint the surfactant flood is the better option than the water flood for a production period of 40 years.

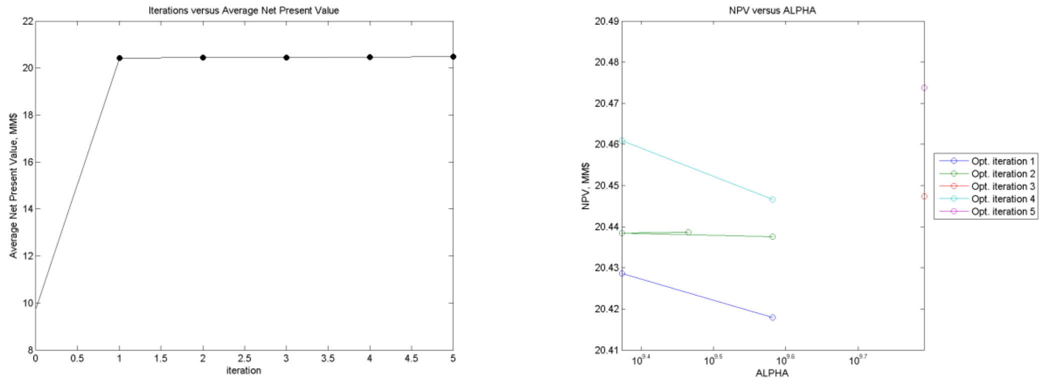
12.3 COMPUTER PROGRAM TIME

The program written for the non-adjoint process and the Monte Carlo sampling using the optimal solution took substantial time to run. Table 20 lists the run times, the number of optimization iterations, the maximum number of weighting factor iterations, and the minimum number of weighting factor iterations.

Table 20: Optimization Time

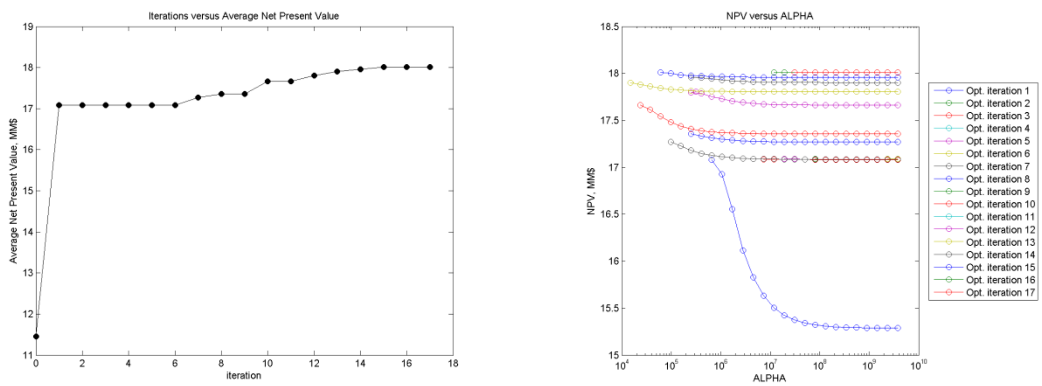
Case	Optimization Iterations	CPU Time, hours	Minimum Alpha iterations	Maximum Alpha iterations
BHP_SURF_20yrs	5	6.4	2	4
BHP_WATER_20yrs	17	24.6	4	28
LQD_SURF_40yrs	3	5.8	2	4
LQD_WATER_40yrs	17	20.3	4	24

The majority of the program time was devoted in determining the weighting factor "alpha" that gave the best average NPV for one optimization iteration. Each weighting factor iteration required 40 ensemble members (40 simulation runs). The optimization iterations along with the iterations to find the optimal alpha can be seen in Figure 87, Figure 88, Figure 89, and Figure 90 for all the cases used in this study.



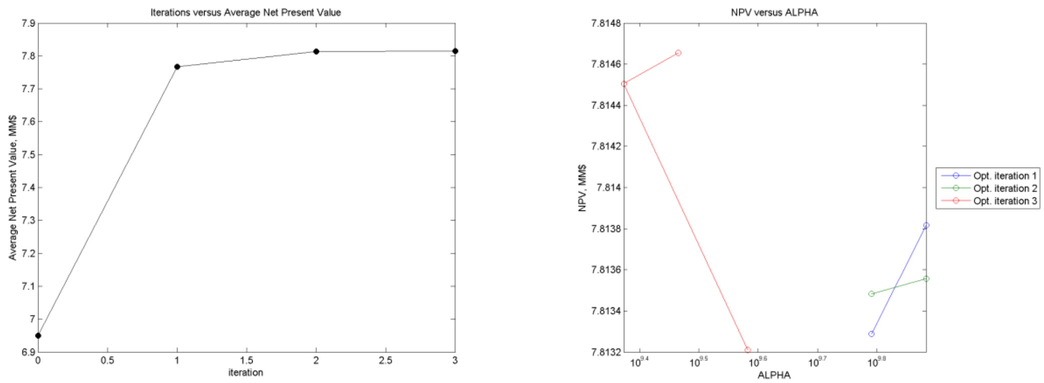
(a) (b)

Figure 87: Optimization Iterations and Weighting Factor Iterations for Surfactant Flood (20 Years): (a) Optimization Iterations, (b) Weighting Factor Iterations



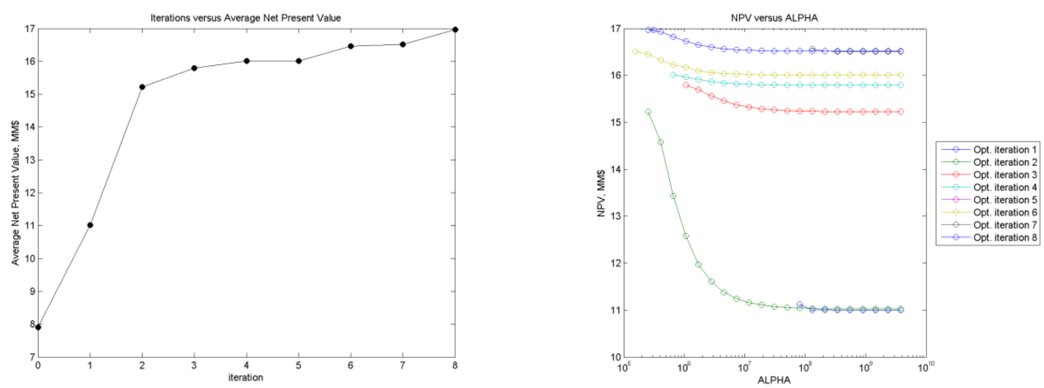
(a) (b)

Figure 88: Optimization Iterations and Weighting Factor Iterations for Water Flood (20 Years): (a) Optimization Iterations, (b) Weighting Factor Iterations



(a) (b)

Figure 89: Optimization Iterations and Weighting Factor Iterations for Surfactant Flood (40 Years): (a) Optimization Iterations, (b) Weighting Factor Iterations



(a) (b)

Figure 90: Optimization Iterations and Weighting Factor Iterations for Water Flood (40 Years): (a) Optimization Iterations, (b) Weighting Factor Iterations

13. CONCLUSIONS

A computer program was developed in Matlab that gives the user the option of optimizing the controls of a surfactant flood for a given period and length of time until a control change. This program gives the user two options for optimizing the producers in addition to the control of the injection rate and concentration of surfactant. These two options are the following:

1. Control of the bottomhole pressure of the producers.
2. Control of the total liquid flow rate of the producers.

Constrained optimization was developed by using a normal score transformation of the ensemble matrix. This transformation bounded the controls with user defined bounds through the use of a uniform distribution of controls and the standard normal distribution.

The created Matlab code has the ability to load realizations of porosity field; to calculate the permeability fields; and to perform an optimization using the mean permeability field, mean porosity field, and the mode of the economic parameters. Once this optimization finished, the program incorporated the optimization solution in a probabilistic setting by using the improved solution to sample every possible scenario of permeability field and economic parameters. The program ultimately generated a cumulative probability distribution of NPV for every possible scenario based on the optimized solution.

13.1 ACCOMPLISHED RESEARCH OBJECTIVES

A summary of the objectives that this research accomplished are the following:

1. Improved the Ensemble Kalman Filter optimization method by improving the weighting factor search by using the Golden Section Search algorithm.
2. Improved the Ensemble Kalman Filter optimization method by constraining the controls using the newly developed normal score transformation of the ensemble matrix.
3. Developed a prior solution based on sound engineering judgment for controls settings (bottomhole pressure control or liquid flow rate control of the producer wells).
4. Wrote a computer program in Matlab that optimizes a surfactant flood using a non-adjoint solution.
5. Defined key factors that impact a surfactant flood in a simulation setting in Schlumberger's Eclipse 100 simulation software package.
6. Developed porosity fields and permeability fields based on observations of porosity and Sequential Gaussian Simulation using Simple Kriging simulation.
7. Developed and verified a program that assimilates realizations of economics and permeability fields using the optimized control vector.

The optimization method used in this research is unique because it does not rely on calculating the adjoints within the simulator. Instead this optimization is done outside of the simulator and thus allows a user to perform optimization using simulators that may not have the ability to calculate adjoints.

13.2 SUMMARY OF RESULTS

The results of this work showed that tightly constrained optimization provides better control of the reservoir parameters while loosely constrained optimization may provide erroneous results due to the poorly bounded controls. In addition to this, it was also learned that an increase in cumulative oil produced due to optimization does not imply an increase in the NPV. Unfavorable costs such as surfactant cost hinder the effect of increased oil production due to optimization. The following are a summary of the results in terms of NPV (Table 21), cumulative oil (Table 22), and recovery factor (Table 23).

Table 21: NPV Optimization Results

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized	Increase in 1st Realization	Increase in Average
BHP_SURF_20yrs	$12,401,997	$9,763,451	$31,298,509	$20,473,815	152.37%	109.70%
BHP_WATER_20yrs	$14,814,147	$11,454,504	$17,598,149	$18,011,221	18.79%	57.24%
LQD_SURF_40yrs	$10,135,908	$6,949,871	$10,266,029	$7,814,655	1.28%	12.44%
LQD_WATER_40yrs	$11,531,566	$7,906,018	$18,232,749	$16,970,953	58.11%	114.66%

Table 22: Cumulative Oil Produced Optimization Results (STB)

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized	Increase in 1st Realization	Increase in Average
BHP_SURF_20yrs	1,059,299	752,526	2,679,722	1,963,373	152.97%	160.90%
BHP_WATER_20yrs	1,051,459	759,922	1,456,869	1,332,572	38.56%	75.36%
LQD_SURF_40yrs	1,047,383	681,510	2,206,360	1,707,843	110.65%	150.60%
LQD_WATER_40yrs	1,045,627	680,587	1,707,625	1,573,665	63.31%	131.22%

Table 23: Recovery Factor Optimization Results (%)

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized
BHP_SURF_20yrs	6.06	4.31	15.33	11.23
BHP_WATER_20yrs	6.02	4.35	8.33	7.62
LQD_SURF_40yrs	5.99	3.90	12.62	9.77
LQD_WATER_40yrs	5.98	3.89	9.77	9.00

13.3 LIMITATIONS OF WORK

There are many limitations in this research. These limitations could have a significant impact on the optimization but the purpose of this research was to develop and test a method for optimization. Therefore, it was imperative to have a simplified reservoir, surfactant flood process, and economic parameters.

One major limitation to this work is the simple reservoir used. A simple reservoir was used to ensure that simulation times were short. For more complex reservoirs, additional simulation time will increase the optimization time. This substantial increase in optimization time will make the method developed in this work archaic when compared to the adjoint method which may converge to a solution more quickly.

Another major limitation to this work is the simplification of the surfactant flood process. Due to the high cost of surfactant, produced surfactant is traditionally re-injected back into the reservoir. This aspect of surfactant flooding was not applied to this work. Re-injecting produced surfactant may decrease the cost of the surfactant flood process. For example, the surfactant produced for the 20 year surfactant flood on the 320 acre field discussed earlier is illustrated in Figure 91.

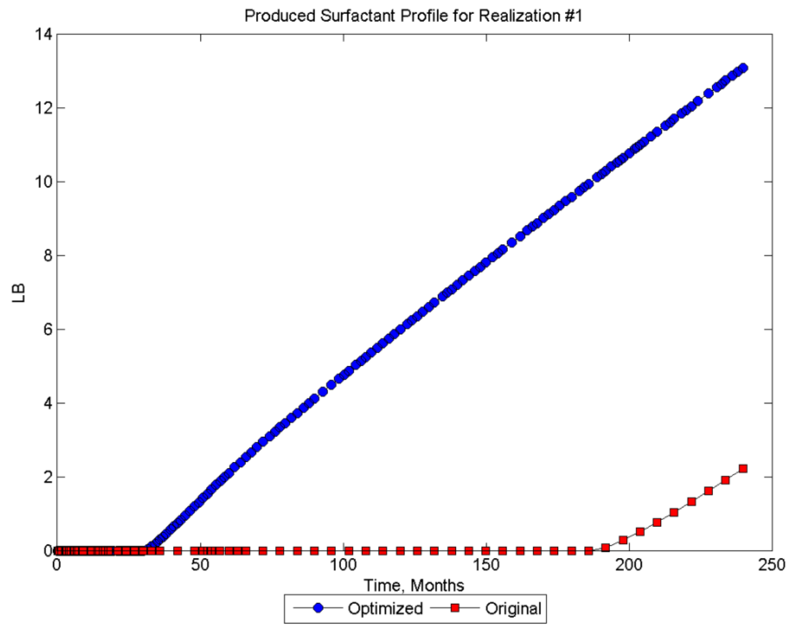


Figure 91: Surfactant Produced for 20 Year Surfactant Flood on 320 Acre Field

It is apparent that there was not much surfactant produced for the 320 acre field (the optimized produced approximately 13 lbs). At $0.80 per pound the surfactant produced resulted in losses of $10.4 for the optimized case. This loss is miniscule and would not affect the optimized controls if the surfactant was re-injected. If the field area was reduced the cost would be much greater. For example, if the field area was reduced to 20 acres Figure 92 would illustrate the surfactant produced.

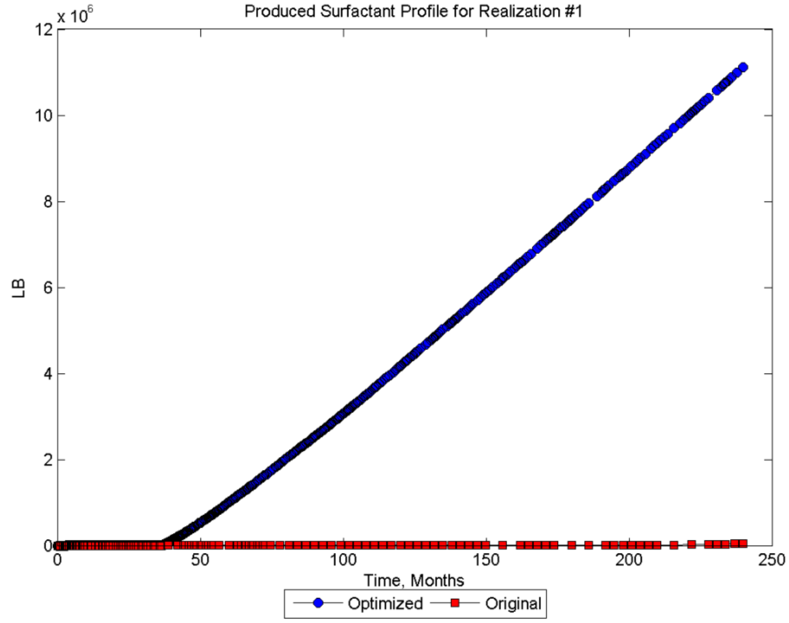


Figure 92: Surfactant Produced for 20 Year Surfactant Flood on 20 Acre Field

From observing the previous figure it can be seen that the surfactant produced is substantial (the optimization resulted in approximately 12 x 10^6 lbs of produced surfactant). If the surfactant was not re-injected into the reservoir, at \$0.80 per pound the surfactant produced would result in losses of \$9.6 million for the optimized case. This loss is significant and warrants extra work to be done in accounting for re-injecting the produced surfactant.

Another limitation to this work is the simplified economic parameters. Each of the economic parameters was constant for the entire life of the project. In real life projects these economic parameters would vary as a function of the oil price. Therefore, future work should account for the economic parameters by having them as a function of the oil price. The oil price can be determined based on an economic forecast of the commodity price of oil. In addition to varying the economic parameters though time, the interest rate should also vary through time based on the market expectation of the company. Market expectation of the company is the potential of the operating company applying the surfactant flood. The interest rate should vary through time based on the forecast of the market expectation of the company.

13.4 FUTURE APPLICATION OF WORK

The impact of this research is that the optimized solution generated is in the form of several realizations of controls. These realizations of optimal controls can be seen as the uncertainty that exist in the optimized solution. In addition to this uncertainty, the cumulative distribution of the 1st realization of controls provides an estimate on the probability of success using the optimal controls. The newly devised constraint system presented in this work can be applied to situations where there is a desire to constrain the controls of the system during optimization. The work flow presented in this work can applied in the optimization of other enhanced oil recovery techniques and different pattern floods. More importantly, this work can be applied to other simulators that are unable to calculate the adjoints.

13.5 ADDENDUM: COMPARISON BETWEEN FIELD SIZE

Well spacing optimization was not the focus of this work but it is important to see the effect of well spacing on the optimized economics of the water flood and surfactant flood. For this short study the acreage was reduced to 20 acres and the same heterogeneous mean porosity field and mean permeability field were used (grid size changed to account for 20 acres). The following DCA parameters (Table 24) and constraints (Table 25) were used.

Table 24: DCA Parameters for Production Time of 20 Years (20 Acre)

	mean	minimum	max
Q_{init}, STB/DAY	500	10	990
b	5	1	9
D_{init}, /year	5	0.1	9.9

Table 25: Constraints for Production Time of 20 Years (20 Acre)

	Minimum	Maximum	Mean	Standard Deviation
Bottom Hole Pressure, PSIA	50	4500	2275	2225
Injection Rate, STB/DAY	10	495000	247505	247495
Surfactant Concentration, LB/STB	0.01	10	5.005	4.995

The following are a summary of the results in terms of NPV (Table 26), cumulative oil (Table 27), and recovery factor (Table 28) for the surfactant flood and water flood for 20 acres using the bottomhole pressure control of the producer wells.

Table 26: NPV Optimization Results for 20 Acres Case

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized
BHP_SURF_20yrs	-$2,446,823	-$576,401	$7,434,775	$7,492,201
BHP_WATER_20yrs	-$2,960,846	-$576,117	$1,279,051	$1,436,658

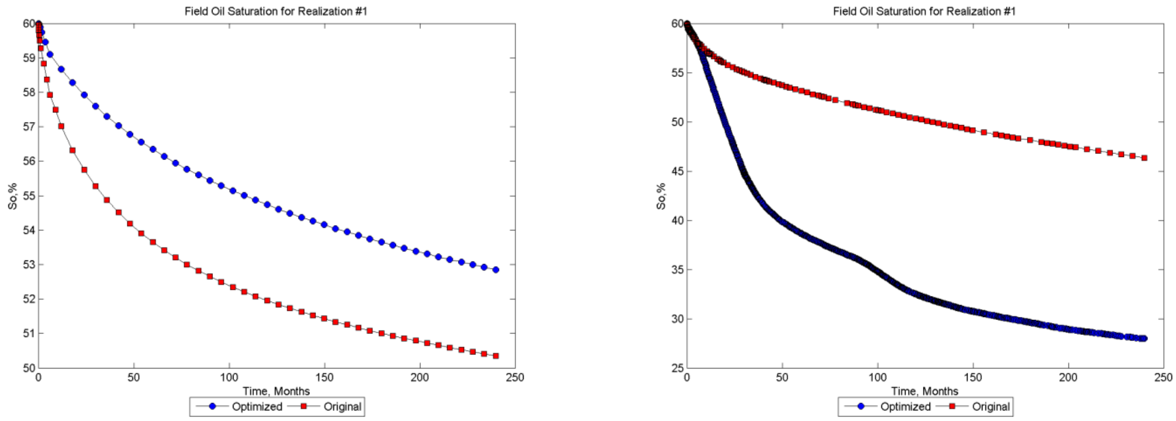
Table 27: Cumulative Oil Produced Optimization Results for 20 Acres (STB)

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized
BHP_SURF_20yrs	247,848	183,378	585,038	518,242
BHP_WATER_20yrs	175,167	149,795	132,820	112,185

Table 28: Recovery Factor Optimization Results for 20 Acres (%)

Case	1st Realization, Original	Average, Original	1st Realization, Optimized	Average, Optimized
BHP_SURF_20yrs	22.69	16.79	53.55	47.44
BHP_WATER_20yrs	16.03	13.71	12.16	10.27

The NPV indicates that the surfactant flood is the better economic choice. The cumulative oil also indicates that the surfactant flood is the better option. The large increases of NPV and cumulative oil for the surfactant flood can be attributed to the large change in the field oil saturation. This observation is conveyed in Figure 93.



(a) (b)

Figure 93: 1st Realization of Field Oil Saturation for 20 Acres: (a) Water Flood, (b) Surfactant Flood

It can be seen that the water flood oil saturation increased from the original to the optimized case. But for the surfactant flood the field oil saturation drops below the water flood residual oil saturation of the field (30%). This shows that the acreage of the flood is important in maximizing the effect of the surfactant flood. The small acreage (20 acres) reduces the oil saturation past the water flood residual oil saturation while the big acreage (320 acres) did not reduce the field oil saturation below the water flood residual oil saturation. Therefore, it can be deduced that surfactant floods need to have small field area to maximize its ability to reduce the field oil saturation. To further support this argument, optimization runs using 20, 40, and 60 acres were done to analyze the effectiveness of different field areas. In all these runs the same DCA parameters, constraints, geological parameters, and economic parameters that were utilized for the 20 acre study were used. The results of this analysis are summarized in Figure 94 and Figure 95 (note the 20 year 320 acre case studied earlier was added to this study).

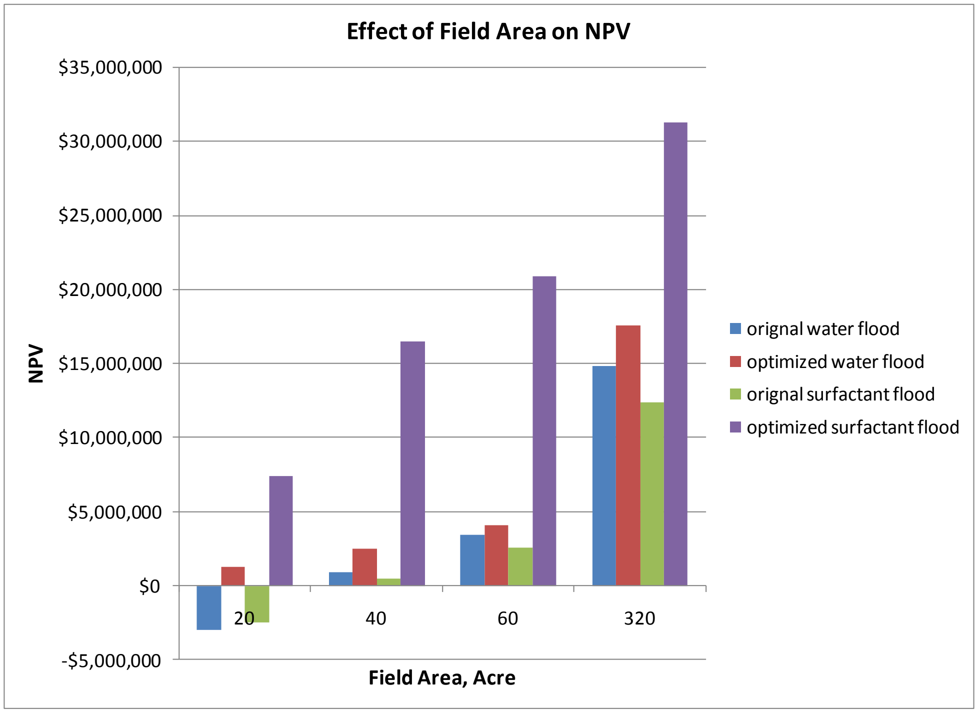


Figure 94: Effect of Field Area on NPV

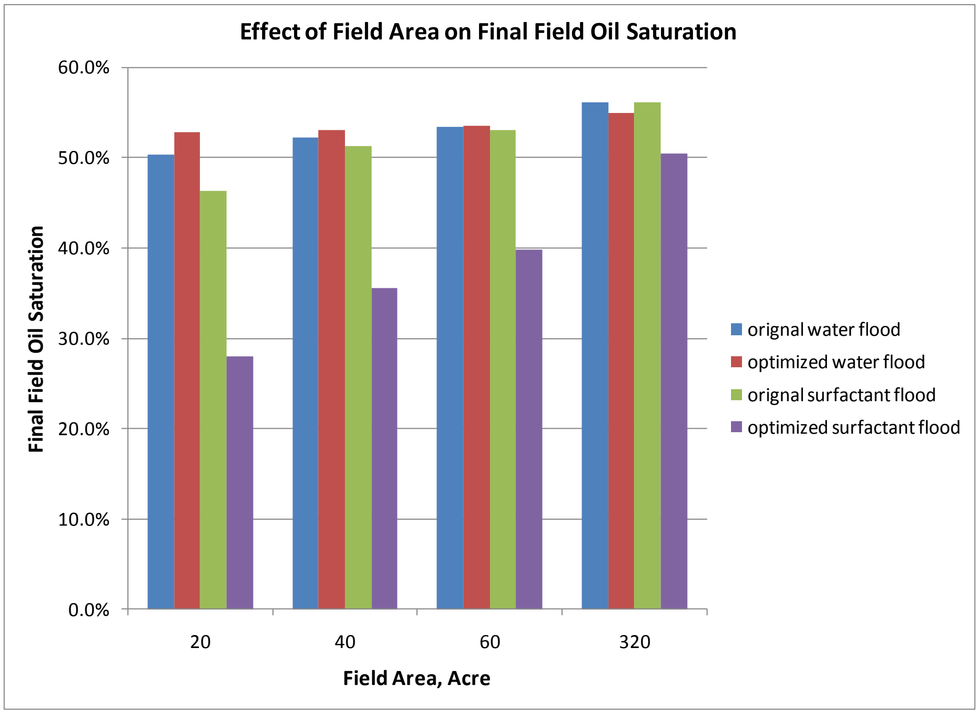


Figure 95: Effect of Field Area on Final Field Oil Saturation

Figure 94 illustrates that as the field area increases the percent increase from the original NPV to the optimized NPV for the water flood and surfactant flood reduces. This is an indication of the difficulty of improving a large area flood. Figure 95 illustrates that for the surfactant flood the oil saturation is minimized for small areas. As a result of this analysis, a small field area must be used to maximize the potential effectiveness of the surfactant flood.

To further illustrate the effectiveness of having a smaller field size, the pore volumes injected for the optimized water flood and surfactant flood show that as the field size decreases the pore volumes injected become larger. Figure 96 and Figure 97 illustrate this observation.

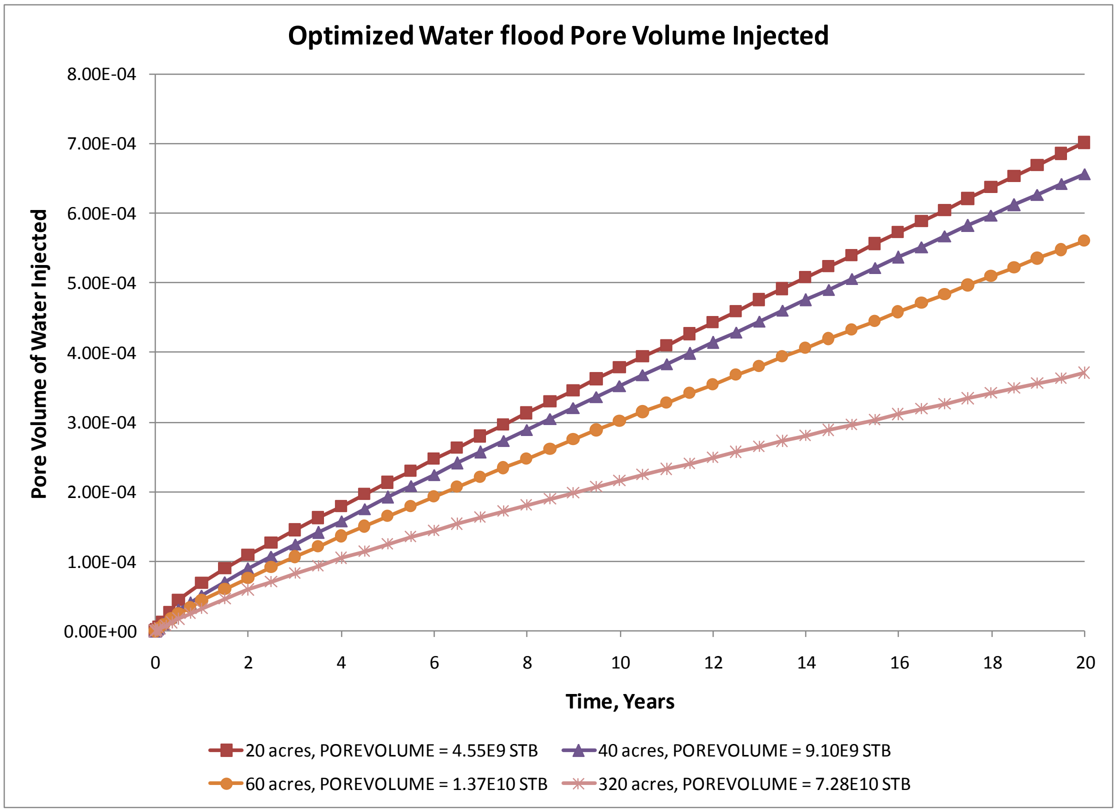


Figure 96: Optimized Water Flood Pore Volume Injected

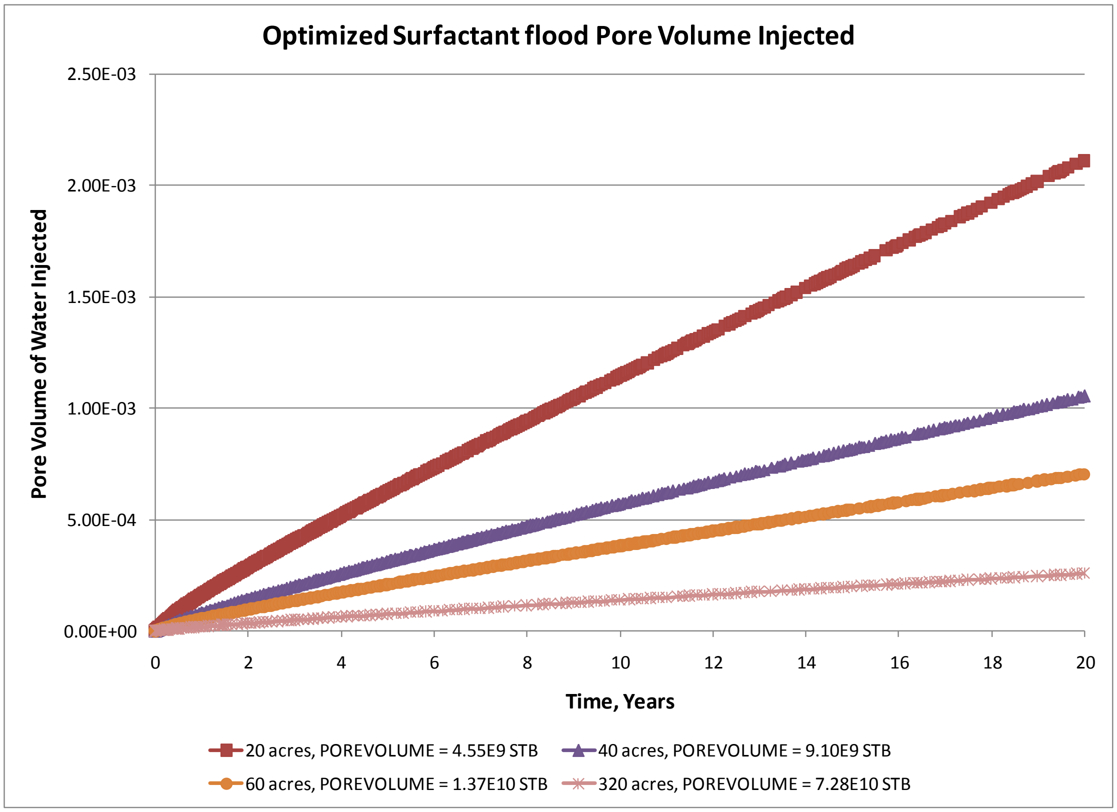


Figure 97: Optimized Surfactant Flood Pore Volume Injected

The previous figures show that it is difficult to increase the pore volume injected for large field areas. This is important because it illustrates that for large pore volumes injected the final oil saturation is minimized (see Figure 95 for final oil saturation plot).

The final conclusion from the field area study is that surfactant floods and water floods need to be utilized for small field areas. It is recommended that future work take this into consideration so that the full potential of the surfactant flood can be achieved.

NOMENCLATURE

Φ= porosity

k = permeability, mD

S_{wir} = irreducible water saturation

k_{ro} = relative permeability of the oil phase

k_{rw} = relative permeability to the water phase

S_w = water saturation

S_{or} = residual oil saturation

n_o = Corey exponent to the oil phase

n_w = Corey exponent to the water phase

k^*_{rw} = endpoint relative permeability to the water phase

k^*_{ro} = endpoint relative permeability to the oil phase

c_o = oil compressibility, psi^{-1}

B_o = oil formation volume factor

P = reservoir pressure, psia

step = current value in the B_o table

B_o^{avg} = average of B_o^{step+1} and B_o^{step}

c_t = total compressibility, psi^{-1}

c_w = water compressibility, psi^{-1}

c_f = formation compressibility, psi^{-1}

S_o = oil saturation

σ = interfacial tension between the oil and water phase

μ = viscosity of the fluid, cp

q = Darcy flow rate, STB

N_c = capillary number

$\nabla\Omega$ = differential of the flow potential

PV = pore volume in the grid block

$CA(S_{CONC})$ = adsorption isotherm , lb/lb

ρ_R = mass density of the rock, lb/RB

M_{abs} = mass of surfactant adsorbed onto the rock, lb

S = objective function

z = nonlinear function

m = vector representing the model variables in reservoir simulator

v = represents the time steps from 0 to V

V = total number of time steps

x = control vector

λ_a = vector of Lagrange multipliers

L = Lagrangian

y = state vector

d = vector containing the simulated data

M = measurement matrix

N_d = number of measurements

N_y = number of variables in the state vector

I = matrix consisting of 1s with dimensions N_d by N_d

C_D = covariance of the data noise

C_Y = covariance of the state vector y for Kalman Filter

y_p = prior estimate of y

G = sensitivity matrix

C_m = covariance matrix of model variables

y_u = update of the estimate of y

d_{obs} = observation data

m_u = update estimate of model

m_p = prior estimate of model

K_e = Kalman gain matrix

$C_{Y,e}$ = covariance of the state vector y for Ensemble Kalman Filter

Ne = number of realizations for Ensemble Kalman Filter

Y = current matrix of ensemble state vectors

Y_p = prior matrix of ensemble state vectors

C_{P1}= Control of production well P1

C_{P2} = Control of production well P2

C_{P3} = Control of production well P3

C_{P4} = Control of production well P4

R_{INJ1} = Injection rate of well INJ1, STB/D

S_{CONC} = Concentration of Surfactant in the Injector, lb/STB

S(x) = objective function

$g(x)$= Net Present Value

α = weighting factor

x_p = prior control vector

C_x = covariance of control vector

δx=incremental control vector change

$F(x'+\delta x)$ = Local quadratic approximation of S(x)

$\gamma = \nabla S(x)$

H = Hessian matrix, $\nabla[\nabla S(x)]^T$

it = iteration number

i = realization number

Ne= total number of realizations

$G(x^{it})$=sensitivity matrix at itth iteration

Y = state matrix

$\overline{Y}$ =average across the rows of the state matrix

y = state vector

NPV = Net present value

M = measurement operator

I = Identity matrix

N_y = number of controls

N_t = number of controls plus one.

c = control

s(c) = the standardization of control *c*

μ_c= mean of control c

σ_c = standard deviation of control c

c_{min} = minimum value of control c

c_{max} = maximum value of control c

Y_s^* = normal score transformation of Y matrix

$\Psi[s(c)]$ = the probability of standardized controller *c* occurring in the normal cdf

R = golden ratio

$alpha_{low}$ = Low guess of alpha

$alpha_{high}$ = High guess of alpha

$alpha_1$ = Low interior point

$alpha_2$ = High interior point

f(alpha) = average NPV across each ensemble

Q_{init} = initial production rate at the start of the production, STB/DAY

D_{init} = initial decline rate at the start of the production

t = time interval through the production life

TSTEP = total number of time intervals

DT = length of the time interval, days

q(t) = total production rate for time interval t, STB

b = Arps decline exponent factor

Min = minimum

Max = maximum

P_i = initial reservoir pressure, psia

BHP = bottomhole pressure, psia

B = formation volume factor

μ = viscosity, cp

h = net pay, ft

r_w = inner radius of the well, ft

J = productivity index, STB/psia

J_o = oil phase productivity index, STB/psia

J_w = water phase productivity index, STB/psia

J_T = total productivity index, STB/psia

C_v = cash flow for time step v (as reported by the simulator)

t_v = time in days corresponding to time step v (as reported by the simulator) v

Δt_n is the change in time step from n-1 to n

RN = random number in the range from 0 to 1

Tr_M = mode of economic parameter

Tr_L = minimum of economic parameter

Tr_H = maximum of economic parameter

WPR = Total water production for time period, STB

SURF = Total amount of surfactant used, lb

WINJ = Total water injected, STB

$\$_{OIL}$ = Oil price

$\$_{Water}$ = Water disposal cost per STB

$\$_{WaterINJ}$ = Water injected cost per STB

$\$_{Surfactant}$ = Surfactant price per lb

C_y = covariance of Y

C_v =total cash flow for time period

r = discount rate, %

R_k = total realizations of permeability fields

R_Φ = total realizations of porosity fields

R_{op} = total realizations of oil price

R_{wi} = total number of realizations of water injection cost

R_{wt} = total number of realizations of water treatment cost

R_{surf} = total number of realizations of surfactant cost

A = area, Acres

BX = horizontal grid block

BY = vertical grid block

GBX = total number of grid blocks in the horizontal direction

GBY = total number of grid blocks in the vertical direction

OOIP = original oil in place, STB

REFERENCES

1. Arroyo-Negrete, E., Datta-Gupta, A., and Devegowda, D. 2006. Streamline-Assisted Ensemble Kalman Filter for Rapid and Continuous Reservoir Model Updating. Paper SPE 104255 presented at the International Oil & Gas Conference and Exhibition, Beijing, China, 5-7 December.
2. Babadagli, T. and Al-Salmi, S. 2002. A Review of Permeability-Prediction Methods for Carbonate Reservoir Using Well-Log Data. Paper SPE 87824 presented at the SPE Asia Pacific Oil and Gas Conference and Exhibition, Melbourne, Australia, 8-10 October.
3. Bardon, C. and Longeron, D.G. 1978. Influence of Very Low Interfacial Tensions on Relative Permeability. Paper SPE 7609 presented at the SPE 53rd Annual Fall Technical Conference and Exhibition, Houston, 1-4 October.
4. Barrufet, M. 2008. Chemical Flooding Notes. PETE 609 Enhanced Oil Recovery Processes, Texas A&M University Department of Petroleum Engineering, Spring Semester.
5. Barrufet, M. 2008. Enhanced Oil Recovery Methods (Chemical, Miscible, Thermal). PETE 609 Enhanced Oil Recovery Processes, Texas A&M University Department of Petroleum Engineering, Spring Semester.
6. Barua, J., Prescott, T., and Haldorsen, H.H. 1986. Financial and Technical Decision Making for Surfactant Flooding. Paper SPE 15074 presented at the 56th California

Regional Meeting of the Socieity of Petroleum Engineers, Oakland, California, 2-4 April.

7. Behrenbruch, P. and Goda, H.M. 2006. Two-Phase Relative Permeability Prediction: A Comparison of the Modified Brooks-Corey Methodology with a New Carman-Kozeny Based Flow Formulation. Paper SPE 101150 presented at the SPE Asia Pacific Oil & Gas Conference and Exhibition, Adelaide, Australia, 11-13 September.
8. Brouwer, D.R. and Jansen, J.D. 2002. Dynamic Optimization of Waterflooding with Smart Wells Using Optimal Control Theory. Paper SPE 78278 presented at the SPE European Petroleum Conference, Aberdeen, U.K., 29-31 October.
9. Chapra, S.C and Canale, R.P. 2002. *Numerical Methods for Engineers: With Software and Programming Applications,* Fourth Edition. New York: McGraw-Hill Higher Education.
10. Chen, Y., Oliver, D.S., and Zhang, D. 2008. Efficient Ensemble-Based Closed-Loop Production Optimization. Paper SPE 112873 presented at the SPE/DOE Improved Oil Recovery Symposium, Tulsa, Oklahoma, 19-23 April.
11. Cinar, Y., Marquez, S., and Orr, F.M. Jr. 2004. Effect of IFT Variation and Wettability on Three-Phase Relative Permeability. Paper SPE 90572 presented at the SPE Annual Technical Conference and Exhibition, Houston, Texas, 26-29 September.
12. Dake, L.P. 2007. *Fundamentals of Reservoir Engineering*. Amsterdam, The Netherlands: Elsevier.

13. Economides, M.J., Hill, A.D., and Ehlig-Economides, C. 1994. *Petroleum Production Systems*. Upper Saddle River, New Jersey: Prentice Hall.

14. Fathi, Z. and Ramirez, W.F. 1982. Optimal Injection Policies for Enhanced Oil Recovery: Part 2—Surfactant Flooding. Paper SPE 12814 retrieved online at http:www.onepetro.org.

15. Fetkovich, M.J., Fetkovich, E.J., and Fetkovich, M.D. 1994. Useful Concepts for Decline-Curve Forecasting, Reserve Estimation, and Analysis. Paper SPE 28628 presented at the SPE Annual Technical Conference & Exhibition, New Orleans, Louisiana, 25-28 September.

16. Gu, Y. and Oliver, D.S. 2005. The Ensemble Kalman Filter for Continuous Updating of Reservoir Simulation Models. *Journal of Energy Resources Technology* **128**: 79-87.

17. Jafarpour, B. 2008. Reservoir Characterization and Forecasting Topic 1: Estimation vs. Simulation. PETE 689 Reservoir Characterization and Forecasting, Texas A&M University Department of Petroleum Engineering, Fall Semester.

18. Jafarpour, B. 2008. Reservoir Characterization and Forecasting Topic 2: Adjoint Method for Gradients. PETE 689 Reservoir Characterization and Forecasting, Texas A&M University Department of Petroleum Engineering, Fall Semester.

19. Jafarpour, B. 2008. Reservoir Characterization and Forecasting Topic 2: Spatial Interpolation and Kriging. PETE 689 Reservoir Characterization and Forecasting, Texas A&M University Department of Petroleum Engineering, Fall Semester.

20. Jafarpour, B. 2008. Reservoir Characterization and Forecasting Topic 3: Review of Probability & Univariate Statistics. PETE 689 Reservoir Characterization and Forecasting, Texas A&M University Department of Petroleum Engineering, Fall Semester.

21. Jafarpour, B. 2008. Reservoir Characterization and Forecasting Topic 3: Variations of Kriging and Cokriging. PETE 689 Reservoir Characterization and Forecasting, Texas A&M University Department of Petroleum Engineering, Fall Semester.

22. Jafarpour, B. 2008. Reservoir Characterization and Forecasting Topic 3: Variogram Estimation and Modeling. PETE 689 Reservoir Characterization and Forecasting, Texas A&M University Department of Petroleum Engineering, Fall Semester 2008.

23. Jakobsen, S.R. and Hovland, F. 1994. Surfactant Flooding: Technical and Economical Conditions to Succeed. Paper SPE 27824 presented at the SPE/DOE Ninth Symposium on Improved Oil Recovery, Tulsa, Oklahoma, 17-20 April.

24. Larson, R.G. 1977. The Influence of Phase Behavior on Surfactant Flooding. Paper SPE 6744 presented at the SPE 52nd Annual Fall Technical Conference and Exhibition, Denver, Colorado, 9-12 October.

25. Lorentzen, R.J., Berg, A.M., Naevdal, G., and Vefring, E.H. 2006. A New Approach for Dynamic Optimization of Waterflooding Problems. Paper SPE 99690 presented at the SPE Intelligent Energy Conference and Exhibition, Amsterdam, The Netherlands, 11-13 April.

26. Mian, M.A. 2002. *Project Economics and Decision Analysis Volume I: Deterministic Models*. Tulsa, Oklahoma: PennWell Corporation.

27. Mian, M.A. 2002. *Project Economics and Decision Analysis Volume II: Probabilistic Models*. Tulsa, Oklahoma: PennWell Corporation.

28. Montgomery, D.C. and Runger, G.C. 2007. *Applied Statistics and Probability for Engineers,* Fourth Edition. Hoboken, New Jersey: John Wiley & Sons.

29. Naevdal, G., Brouwer, D.R., and Jansen, J. 2006. Waterflooding Using Closed-loop Control. *Computational Geosciences* **10**: 37-60.

30. Nwaozo, J. 2006. Dynamic Optimization of a Water Flood Reservoir. M.S. thesis, Norman, Oklahoma: University of Oklahoma.

31. Porzucek, C. and Ramirez, W.F. 1988. Optimal Injection Strategies for Surfactant Flooding Enhanced Oil Recovery Using a Streamtube Reservoir Simulator—Part 1, The Simulator. Paper SPE 17095 retrieved online at http:www.onepetro.org.

32. Porzucek, C. and Ramirez, W.F. 1988. Optimal Injection Strategies for Surfactant Flooding Enhanced Oil Recovery Using a Streamtube Reservoir Simulator—Part 2, The Optimal Control Problem. Paper SPE 17096 retrieved online at http:www.onepetro.org.

33. Porzucek, C. and Ramirez, W.F. 1988. Optimal Injection Strategies for Surfactant Flooding Enhanced Oil Recovery Using a Streamtube Reservoir Simulator—Part 3, Results. Paper SPE 17097 retrieved online at http:www.onepetro.org.

34. Rao, D.N., Girad, M., and Sayogh, S.G. 1990. Impact of Miscible Flooding on Wettability, Relative Permeability, and Oil Recovery. Paper SPE 20522 presented at the SPE Annual Technical Conference and Exhibition, New Orleans, Louisiana, 23-26 September.

35. Recktenwald, G. 1995. Matlab program: readColData.m, Program retrieved from http://web.cecs.pdx.edu/~gerry/MATLAB/plotting/examples/readColData.m.

36. Remy, N. 2004. *Geostatistical Modeling Software: User's Manual*, Stanford, California.

37. Sarma, P., Aziz, K., and Durlofsky, L.J. 2005. Implementation of Adjoint Solution for Optimal Control of Smart Wells. Paper SPE 92864 presented at the SPE Reservoir Simulation Symposium, Houston, Texas, 31 January-2 Febuary.

38. Schlumberger. 2008. *Eclipse Reference Manual*, Houston, Texas.

39. Schlumberger. 2008. *Eclipse Technical Description Manual*, Houston, Texas.

40. Sun, Y. 2009. Petroleum Geology Lecture 17: Frontiers in Reservoir Research. GEOL 603 Petroleum Geology, Texas A&M University Department of Geology & Geophysics, Spring Semester.

41. Taber, J.J., Martin, F.D. and Seright, R.S. 1996. EOR Screening Criteria Revisted—Part 1: Introduction to Screening Criteria and Enhanced Recovery Field Projects. Paper SPE 35385 presented at the SPE/DOE Improved Oil Recovery Symposium, Tulsa, Oklahoma, 21-24 April.

42. Taber, J.J., Martin, F.D. and Seright, R.S. 1996. EOR Screening Criteria Revisted—Part 2: Applications and Impact of Oil Prices. Paper SPE 39234 presented at the SPE/DOE Improved Oil Recovery Symposium, Tulsa, Oklahoma, 21-24 April.

43. Wu, W., Vaskas, A., Delshad, M., and Sepehrnoori, K. 1996. Design and Optimization of Low-Cost Chemical Flooding. Paper SPE 35355 presented at the

SPE/DOE Tenth Symposium on Improved Oil Recovery, Tulsa, Oklahoma, 21-24 April.

44. Zandvliet, M.J., Handels, M., van Essen, G.M, Brouwer, D.R., and Jansen, J.D. 2007. Adjoint-Based Well-Placement Optimization Under Production Constraints. Paper SPE 105797 presented at the Reservoir Simualtion Symposium, Houston, Texas, 26-28 February.

45. Zhang, J., Delshad, M., Sepehrnoori, K. and Pope, G.A. 2005. An Efficient Reservoir Simulation Approach to Design and Optimize Improved Oil Recovery Processes with Distributed Computing. Paper SPE 94733 presented at the SPE Latin American and Caribbean Petroleum Engineering Conference, Rio de Janeiro, Brazil, 20-23 June.

46. Zhoe, D. and Stenby, E.H. 1993. Displacement of Trapped Oil From Water-Wet Reservoir Rock. *Transport in Porous Media* **11**: 1-16.

APPENDIX

ECLIPSE DATA SIMULATION FILE

The data file created using this research was based partly on the example surfactant flood simulation file given in eclipse. The "ALL.IN" section of this data contains all the information of the controls for all timesteps; these controls are written from Matlab. Also "ALL.IN" contains the location and dimensions of the wells in the 5 spot configuration.

SURFACT.DATA

```
RUNSPEC
TITLE
 Surfactant model test case.

DIMENS
   50   50    1  /

OIL

WATER

SURFACT

FIELD

TABDIMS
    2    1   20   20    1   20 /

WELLDIMS
 6 7 3 6 5 10 5 4 3 0 1 1 /


START
   1 'JAN' 1983  /

NSTACK
    1000 /



GRID      ==============================================================

PSEUDO

GRID

EQUALS
'DX' 74.6705/
'DY' 74.6705/
'DZ' 70 1 50 1 50 1 1/
'TOPS' 10000/
```

```
/

--EQUALS
INCLUDE
'PORO.IN' /


INCLUDE
'PERM.IN' /



COPY
PERMX PERMY/
PERMX PERMZ/
/

MULTIPLY
PERMZ .2/
/




PROPS     ===============================================================

INCLUDE
'SWFN.IN' /

INCLUDE
'SOF2.IN' /



PVTW
  4500 1 3E-6 1 0.0  /

PVDO
  50       1.00001     3
  4500     1.00000     3
  10000    0.99998     3
/

ROCK
 4500 5E-6 /

DENSITY
 56 62.4 .062428 /

INCLUDE
'SURFACTALL.IN' /



RPTPROPS
    18*0 1 /

REGIONS    ==============================================================

SATNUM
 2500*1 /

SURFNUM
 2500*2 /

RPTREGS
```

```
 24*0 /

SOLUTION   ==============================================================



INCLUDE
'SWAT.IN'/

INCLUDE
'PRESSURE.IN'/



RPTSOL
1 1 1 0 0   0 1 0 0 0
0 0 0 0 0   0 0 0 0 0
0 0 0 0 0   0 0 0 0 0
1 2 1 0 0   0 0 2 0 0
0 0 0 2 0   2 0 0 0 0 /

SUMMARY    ============================================================

EXCEL

WBHP
/
FWIR
FOPR
--FOPT
FPR
FTPRSUR

FOE
FTIRSUR
----FTITSUR
FOSAT
WTPRSUR
'OP' /

FWPR
--FWPT
FGPR
FWCT
--FWIT
FWSAT
FWIR
WWCT
/

WOPT
/

WOPR
/

WLPR
/




RUNSUM

SCHEDULE   ============================================================

TUNING
```

```
/
/
2* 1000/







INCLUDE
'ALL.IN'/

END
```

Wflood_sch.INC

```
ECHO

WELSPECS
'P1' '5SPOTP' 1 1 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'P2' '5SPOTP' 1 50 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'P3' '5SPOTP' 50 50 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'P4' '5SPOTP' 50 1 10035 'LIQ' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/

WELSPECS
'INJ1' 'INJECTOR' 25 25 10035 'WATER' 1* 'STD' 'SHUT' 'YES' 1* 'SEG' 3* 'STD' /
/


COMPDAT
'P1' 1 1 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'P2' 1 50 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'P3' 50 50 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'P4' 50 1 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/

COMPDAT
'INJ1' 25 25 1 1 'OPEN' 2* .8 3* 'Z' 1* /
/
```

MATLAB PROGRAM

The Matlab program written for this research incorporates the SURFACT.DATA file in the optimization of NPV with respect to the controls in the five spot and the assimilation of probability. The Main Program calls all the neccassary routines needed to perform the optimization and the Monte Carlo sampling. To read the RSM file for some of the subroutines a file written by Gerald Recktenwald (Recktenwald, 1995) was used.

MAIN PROGRAM

This is the main program that defines the optimization settings and Monte Carlo settings.

NPV_algorithbeta1.m

```
clear; % clear all variables from memory
tic;
runs=1;
for rNe=1:runs
close all; % close all figure windows
fclose('all'); %close all files associated with eclipse
rand('twister',5489);
NR=10; % Number of realizations of price, permeability, porosity realizations
%% Reservoir Properties

relperm_o=1;
S_w=.40;

FVF_o=1;
FVF_w=1;
visc_o=3; %cp
visc_w=1; %cp
visc_a=(visc_o+visc_w)/2;
pay=70; %ft
density_o=56; %lb/ft^3
density_w=62.4; %lb/ft^3

cf=5E-6; %(1/psi)
cw=3E-6; %(1/psi)
co=2.2472E-9; %(1/psi)
rw=.8; %ft

Pi=4500; %psi.
Area=320; %Acre
% Area=320; %Acre
density_r=1000; %kg/m^3
%% Surfactant Properties and Miscible Properties
SURFVISC_IM=[0,1;10.516,5];
```

```
SURFVISC_M=[0,1;10.516,5];
SURFADS_IM=[0,0;.350533,.0005;10.516,.0005];
SURFADS_M=[0,0;.350533,.0005;10.516,.0005];
SURFST_IM=[0,.05*.00571;.350533,.00571*1E-6;10.516,.00571*1E-6];
SURFST_M=[0,.05*.00571;.350533,.00571*1E-6;10.516,.00571*1E-6];
SURFCAPD_IM=[-9,0;-4.5,0;-2,1;10,1];
SURFCAPD_M=[-9,0;-4.5,0;-2,1;10,1];
SURFROCK=[1,1000;1,1000];
Sorm_min=.01;
Swrm_min=.1;
Swrm_max=1-Sorm_min;
Sorm_max=1-Swrm_min;
krom_min=0;
krom_max=1;
krwm_min=0;
krwm_max=1;
Abs_ISOTHERM=SURFADS_IM(:,2);
Conc_ISOTHERM=SURFADS_IM(:,1);
SURFACTALL(SURFVISC_IM,SURFVISC_M,SURFADS_IM,SURFADS_M,SURFST_IM,SURFST_M,SURFCAPD_IM,SUR
FCAPD_M,SURFROCK)
%% Porosity, Permeability, Capillary, Immiscible Properties

POROALL=load('PORO_NS1');
Swir=.1;
Sor=.3;
kroe=1;
krwe=1;
no=6;
nw=4;
Pd=.145038; %psi
PRS_INDEX=.5;
TLENGTH=12;





for r=1:2500
    for c=1:length(POROALL(1,:))
        PERMALL(r,c)=(100*(POROALL(r,c)^2)*(1-Swir)/Swir)^2;
    end

end



for t=1:2500 %To generate vector that contains the means of each row of the realizations
  PORO_MEAN(t,1)=mean(POROALL(t,:));
  PERM_MEAN(t,1)=geomean(PERMALL(t,:));
end



[PERM_INIT,Perm1,Perm2,Perm3,Perm4,Perm,PORO_INIT,por1,por2,por3,por4,por,Sw,So,kro,krw,P
c]=perm_por(PERM_MEAN,PORO_MEAN,S_w,Pi,Swir,Sor,kroe,krwe,no,nw,Pd,PRS_INDEX,TLENGTH,Sorm
_min,Sorm_max,Swrm_min,Swrm_max,krom_min,krom_max,krwm_min,krwm_max);

%% Region OIP and Field OIP calculation
for inc=1:length(PORO_MEAN)
 OIP_BLOCK(inc) = 7758*PORO_MEAN(inc)*(1-S_w)*pay*Area/(FVF_o*50^2); %STB
 Area_BLOCK(inc)=Area/50^2;%Acre
 PV_vector(inc)=7758*pay*Area*PORO_MEAN(inc);

end
PVTOTAL=sum(PV_vector); %STB
OIP_GRID=reshape (OIP_BLOCK, 50, 50)';
OIP_GRID1=OIP_GRID(1:25,1:25);
```

```
OIP1=sum(reshape(OIP_GRID1,25^2,1));
OIP_GRID2=OIP_GRID(26:50,1:25);
OIP2=sum(reshape(OIP_GRID2,25^2,1));
OIP_GRID3=OIP_GRID(26:50,26:50);
OIP3=sum(reshape(OIP_GRID3,25^2,1));
OIP_GRID4=OIP_GRID(1:25,26:50);
OIP4=sum(reshape(OIP_GRID4,25^2,1));
OIP = sum(OIP_BLOCK);
Area_GRID=reshape (Area_BLOCK, 50, 50);
Area_GRID1=Area_GRID(1:25,1:25);
Area1=sum(reshape(Area_GRID1,25^2,1));
Area_GRID2=Area_GRID(26:50,1:25);
Area2=sum(reshape(Area_GRID2,25^2,1));
Area_GRID3=Area_GRID(26:50,26:50);
Area3=sum(reshape(Area_GRID3,25^2,1));
Area_GRID4=Area_GRID(1:25,26:50);
Area4=sum(reshape(Area_GRID4,25^2,1));


%% Prices and Costs parameters
op_pmode=70; %Mode Price of oil $/STB
op_pmin=20; %Min Price of oil $/STB
op_pmax=120; %Max Price of oil $/STB

for q=1:NR
    roilprice(q)=rand;
    if roilprice(q)<=(op_pmode-op_pmin)/(op_pmax-op_pmin)
       op_preal(q)=op_pmin+((op_pmode-op_pmin)*(op_pmax-op_pmin)*roilprice(q))^(1/2);
    end
    if roilprice(q)>=(op_pmode-op_pmin)/(op_pmax-op_pmin)
       op_preal(q)=op_pmax-((op_pmax-op_pmode)*(op_pmax-op_pmin)*(1-roilprice(q)))^(1/2);
    end
end

wp_pmode=2; %Mode water disposal $/STB
wp_pmin=1; %Min water disposal $/STB
wp_pmax=3; %Max water disposal $/STB
for q=1:NR
    wpprice(q)=rand;
    if wpprice(q)<=(wp_pmode-wp_pmin)/(wp_pmax-wp_pmin)
       wp_preal(q)=wp_pmin+((wp_pmode-wp_pmin)*(wp_pmax-wp_pmin)*wpprice(q))^(1/2);
    end
    if wpprice(q)>=(wp_pmode-wp_pmin)/(wp_pmax-wp_pmin)
       wp_preal(q)=wp_pmax-((wp_pmax-wp_pmode)*(wp_pmax-wp_pmin)*(1-wpprice(q)))^(1/2);
    end
end

winj_pmode=1; %Mode Price of injected water $/STB
winj_pmin=.7; %Min Price of injected water $/STB
winj_pmax=2; %Max Price of injected water $/STB
for q=1:NR
    winjprice(q)=rand;
    if winjprice(q)<=(winj_pmode-winj_pmin)/(winj_pmax-winj_pmin)
       winj_preal(q)=winj_pmin+((winj_pmode-winj_pmin)*(winj_pmax-
winj_pmin)*winjprice(q))^(1/2);
    end
    if winjprice(q)>=(winj_pmode-winj_pmin)/(winj_pmax-winj_pmin)
       winj_preal(q)=winj_pmax-((winj_pmax-winj_pmode)*(winj_pmax-winj_pmin)*(1-
winjprice(q)))^(1/2);
    end
end

surf_pmode=1; %Mode Price of surfactant used $/lb
surf_pmin=.8; %Min Price of surfactant used $/lb
surf_pmax=1.5; %Max Price of surfactant used $/lb
for q=1:NR
    surfprice(q)=rand;
```

```
    if surfprice(q)<=(surf_pmode-surf_pmin)/(surf_pmax-surf_pmin)
       surf_preal(q)=surf_pmin+((surf_pmode-surf_pmin)*(surf_pmax-
surf_pmin)*surfprice(q))^(1/2);
    end
    if surfprice(q)>=(surf_pmode-surf_pmin)/(surf_pmax-surf_pmin)
       surf_preal(q)=surf_pmax-((surf_pmax-surf_pmode)*(surf_pmax-surf_pmin)*(1-
surfprice(q)))^(1/2);
    end
end

op_p=op_pmode; %Price of oil $/STB
wp_p=wp_pmode; %Price of water disposal $/STB
winj_p=winj_pmode; %Price of injected water $/STB
surf_p=surf_pmode; %Price of surfactant used $/lb
r=10; %Discount rate in %

%% Optimization Settings
PCON=2; %2=Liquid FLOWRATE CONTROL OF PRODUCERS / 1=Oil FLOWRATE CONTROL OF PRODUCERS /
0=BHP CONTROL OF PRODUCERS
TSTEP=40;
DT=365/2;
TIMESTEPS=TSTEP;
Nt=6*TSTEP;
Nd=1;
Ny=Nt+1;


ALPHAinit= 1E20;
DALPHA=5E5;
ALPHAlow=1E3;
ALPHAhigh=1E10;
maxit=100;
es=50;
ec=1E-4;

Z=zeros(Nd,Ny-Nd);
I=eye(Nd,Nd);
M=[Z,I];
Ne=40; %Number of MonteCarlo realizations of controls
Graph=1; %Number of output graphs of realizations

%% Control Parameters

Qinmean=500; %STB/Day
Qinmin=10;  %STB/Day
Qinmax=2*Qinmean-Qinmin;  %STB
Dinmean=5; %1/year
Dinmin=.1;  %1/year
Dinmax=2*Dinmean-Dinmin;  %1/year
bmean=5;
bmin=1;
bmax=2*bmean-bmin;
CmuC=.350533; %lb/STB

IFT=1E-6; %N/m
TRANSFORM=1; %TRANSFORM=0 for ln transform; TRANSFORM=1 for gaussian transform;
if PCON==0
BHPmin=50;
BHPmax=Pi;
CP1min=BHPmin;
CP1max=BHPmax;
CP2min=BHPmin;
CP2max=BHPmax;
CP3min=BHPmin;
CP3max=BHPmax;
CP4min=BHPmin;
CP4max=BHPmax;
```

```
end
if PCON==2
LQDmin=10;
LQDmax=Qinmax;
CP1min=LQDmin;
CP1max=LQDmax;
CP2min=LQDmin;
CP2max=LQDmax;
CP3min=LQDmin;
CP3max=LQDmax;
CP4min=LQDmin;
CP4max=LQDmax;
end
RINJ1min=Qinmin;
RINJ1max=Qinmax*4;
CONCmin=.01;
CONCmax=10;
% CONCmin=0;
% CONCmax=0;
CP1mu=(CP1min+CP1max)/2;
CP1stdv=(CP1max-CP1min)/2;
CP2mu=(CP2min+CP2max)/2;
CP2stdv=(CP2max-CP2min)/2;
CP3mu=(CP3min+CP3max)/2;
CP3stdv=(CP3max-CP3min)/2;
CP4mu=(CP4min+CP4max)/2;
CP4stdv=(CP4max-CP4min)/2;
RINJmu=(RINJ1max+RINJ1min)/2;
RINJstdv=(RINJ1max-RINJ1min)/2;
CONCmu=(CONCmax+CONCmin)/2;
CONCstdv=(CONCmax-CONCmin)/2;
% CONCstdv=1;
%% Period calculations

for t=1:TSTEP
    period(t)=t*DT/(365/12);
end

%% Prior Matrix Construction
for i=1:Ne

    Qin1(i)=Qinmin+rand*(Qinmax-Qinmin);
    Din1(i)=Dinmin+rand*(Dinmax-Dinmin);
    b1(i)=bmin+rand*(bmax-bmin);

    Qin2(i)=Qinmin+rand*(Qinmax-Qinmin);
    Din2(i)=Dinmin+rand*(Dinmax-Dinmin);
    b2(i)=bmin+rand*(bmax-bmin);

    Qin3(i)=Qinmin+rand*(Qinmax-Qinmin);
    Din3(i)=Dinmin+rand*(Dinmax-Dinmin);
    b3(i)=bmin+rand*(bmax-bmin);

    Qin4(i)=Qinmin+rand*(Qinmax-Qinmin);
    Din4(i)=Dinmin+rand*(Dinmax-Dinmin);
    b4(i)=bmin+rand*(bmax-bmin);


    for t=1:TSTEP
        grp(t)=6*t;

        if t==1

            krofieldp(t,i)=interp1([0,Sw,1],[1,kro,0],S_w);
            krwfieldp(t,i)=interp1([0,Sw,1],[0,krw,1],S_w);

            Sw1p(t,i)=S_w;
```

```
            Sw2p(t,i)=S_w;
            Sw3p(t,i)=S_w;
            Sw4p(t,i)=S_w;
            kro1p(t,i)=krofieldp(t,i);
            kro2p(t,i)=krofieldp(t,i);
            kro3p(t,i)=krofieldp(t,i);
            kro4p(t,i)=krofieldp(t,i);
            krw1p(t,i)=krwfieldp(t,i);
            krw2p(t,i)=krwfieldp(t,i);
            krw3p(t,i)=krwfieldp(t,i);
            krw4p(t,i)=krwfieldp(t,i);
            ct(t,i)=S_w*cw+(1-S_w)*co+cf;
            Swfieldp(t,i)=S_w;
            Pi_fieldp(t,i)=Pi;
            FTOTSURp(t,i)=0;
            SINJrate(t,i)=0;
            PVcurrent(t,i)=PVTOTAL;
            FLRp(t,i)=0;


        end




        q1(t,i)=Qin1(i)*(1+b1(i)*Din1(i)*DT*(t-1)/365)^(-1/b1(i));

        q2(t,i)=Qin2(i)*(1+b2(i)*Din2(i)*DT*(t-1)/365)^(-1/b2(i));

        q3(t,i)=Qin3(i)*(1+b3(i)*Din3(i)*DT*(t-1)/365)^(-1/b3(i));

        q4(t,i)=Qin4(i)*(1+b4(i)*Din4(i)*DT*(t-1)/365)^(-1/b4(i));

        qmean(t,i)=Qinmean*(1+bmean*Dinmean*DT*(t-1)/365)^(1/-bmean);




        if PCON==0
            PI_1(t,i)=((Perm1*kro1p(1,i)*pay)/(162.6*FVF_o*visc_o))*((log10(DT*(t-
1)*24+1)+log10(Perm1*kro1p(1,i)/(por1*visc_o*ct(1,i)*rw^2))-3.23)^-
1)+((Perm1*krw1p(1,i)*pay)/(162.6*FVF_w*visc_w))*((log10(DT*(t-
1)*24+1)+log10(Perm1*krw1p(1,i)/(por1*visc_w*ct(1,i)*rw^2))-3.23)^-1);
            PI_2(t,i)=((Perm2*kro2p(1,i)*pay)/(162.6*FVF_o*visc_o))*((log10(DT*(t-
1)*24+1)+log10(Perm2*kro2p(1,i)/(por2*visc_o*ct(1,i)*rw^2))-3.23)^-
1)+((Perm2*krw2p(1,i)*pay)/(162.6*FVF_w*visc_w))*((log10(DT*(t-
1)*24+1)+log10(Perm2*krw2p(1,i)/(por2*visc_w*ct(1,i)*rw^2))-3.23)^-1);
            PI_3(t,i)=((Perm3*kro3p(1,i)*pay)/(162.6*FVF_o*visc_o))*((log10(DT*(t-
1)*24+1)+log10(Perm3*kro3p(1,i)/(por3*visc_o*ct(1,i)*rw^2))-3.23)^-
1)+((Perm3*krw3p(1,i)*pay)/(162.6*FVF_w*visc_w))*((log10(DT*(t-
1)*24+1)+log10(Perm3*krw3p(1,i)/(por3*visc_w*ct(1,i)*rw^2))-3.23)^-1);
            PI_4(t,i)=((Perm4*kro4p(1,i)*pay)/(162.6*FVF_o*visc_o))*((log10(DT*(t-
1)*24+1)+log10(Perm4*kro4p(1,i)/(por4*visc_o*ct(1,i)*rw^2))-3.23)^-
1)+((Perm4*krw4p(1,i)*pay)/(162.6*FVF_w*visc_w))*((log10(DT*(t-
1)*24+1)+log10(Perm4*krw4p(1,i)/(por4*visc_w*ct(1,i)*rw^2))-3.23)^-1);
            PI_mean(t,i)=((Perm*krofieldp(1,i)*pay)/(162.6*FVF_o*visc_o))*((log10(DT*(t-
1)*24+1)+log10(Perm*krofieldp(1,i)/(por*visc_o*ct(1,i)*rw^2))-3.23)^-
1)+((Perm*krwfieldp(1,i)*pay)/(162.6*FVF_w*visc_w))*((log10(DT*(t-
1)*24+1)+log10(Perm*krwfieldp(1,i)/(por*visc_w*ct(1,i)*rw^2))-3.23)^-1);
```

```
            BHP1(t,i)=Pi_fieldp(1,i)-q1(t,i)/PI_1(t,i);
            BHP2(t,i)=Pi_fieldp(1,i)-q2(t,i)/PI_2(t,i);
            BHP3(t,i)=Pi_fieldp(1,i)-q3(t,i)/PI_3(t,i);
            BHP4(t,i)=Pi_fieldp(1,i)-q4(t,i)/PI_4(t,i);
            BHPmean(t,i)=Pi_fieldp(1,i)-qmean(t,i)/PI_mean(t,i);


              RINJ1(t,i)=((PI_1(t,i)*(Pi_fieldp(1,i)-BHP1(t,i))+PI_2(t,i)*(Pi_fieldp(1,i)-
BHP2(t,i))+PI_3(t,i)*(Pi_fieldp(1,i)-BHP3(t,i))+PI_4(t,i)*(Pi_fieldp(1,i)-BHP4(t,i))));
              RINJ1mean(t,i)=(4*PI_mean(t,i)*(Pi_fieldp(1,i)-BHPmean(t,i)));

        end

        if PCON==1


        qo1(t,i)=q1(t,i);
        qo2(t,i)=q2(t,i);
        qo3(t,i)=q3(t,i);
        qo4(t,i)=q4(t,i);
        qomean(t,i)=qmean(t,i);

        RINJ1(t,i)=q1(t,i)+q2(1,i)+q3(t,i)+q4(t,i);
        RINJ1mean(t,i)=4*qmean(t,i);


        end

        if PCON==2

           RINJ1(t,i)=(q1(t,i)+q2(t,i)+q3(t,i)+q4(t,i));
           RINJ1mean(t,i)=(4*qmean(t,i));


        end

        CONC(t,i)=CmuC;
        CONCmean(t,i)=CmuC;





        if PCON==2
            CP1(t,i)=q1(t,i);
            CP2(t,i)=q2(t,i);
            CP3(t,i)=q3(t,i);
            CP4(t,i)=q4(t,i);
            CPmean(t,i)=qmean(t,i);



        end

        if PCON==1
            CP1(t,i)=qo1(t,i);
            CP2(t,i)=qo2(t,i);
            CP3(t,i)=qo3(t,i);
            CP4(t,i)=qo4(t,i);
            CPmean(t,i)=qomean(t,i);




        end
```

```
        if PCON==0
            CP1(t,i)=BHP1(t,i);
            CP2(t,i)=BHP2(t,i);
            CP3(t,i)=BHP3(t,i);
            CP4(t,i)=BHP4(t,i);
            CPmean(t,i)=BHPmean(t,i);






        end

        if t>1
                        xt(grp(t-
1)+1:grp(t),i)=[CP1(t,i);CP2(t,i);CP3(t,i);CP4(t,i);RINJ1(t,i);CONC(t,i)];
                        else
xt(t:grp,i)=[CP1(t,i);CP2(t,i);CP3(t,i);CP4(t,i);RINJ1(t,i);CONC(t,i)];
        end






    end
    [NPVinit(i),
Poilinit(i)]=surfblackbox1(TSTEP,DT,grp,i,xt,PCON,op_p,wp_p,winj_p,surf_p,r);

end

%% Transform xt to x transformed matrix
for i=1:Ne
    for t=1:TSTEP
        if TRANSFORM==0
                    if t>1
                        x(grp(t-
1)+1:grp(t),i)=[log(CP1(t,i));log(CP2(t,i));log(CP3(t,i));log(CP4(t,i));log(RINJ1(t,i));l
og(CONC(t,i))];
                        else
x(t:grp,i)=[log(CP1(t,i));log(CP2(t,i));log(CP3(t,i));log(CP4(t,i));log(RINJ1(t,i));log(C
ONC(t,i))];
                    end
        end

        if TRANSFORM==1
                    if t>1
                        x(grp(t-1)+1:grp(t),i)=[(CP1(t,i)-CP1mu)/CP1stdv;(CP2(t,i)-
CP2mu)/CP2stdv;(CP3(t,i)-CP3mu)/CP3stdv;(CP4(t,i)-CP4mu)/CP4stdv;(RINJ1(t,i)-
RINJmu)/RINJstdv;(CONC(t,i)-CONCmu)/CONCstdv];
                        else x(t:grp,i)=[(CP1(t,i)-CP1mu)/CP1stdv;(CP2(t,i)-
CP2mu)/CP2stdv;(CP3(t,i)-CP3mu)/CP3stdv;(CP4(t,i)-CP4mu)/CP4stdv;(RINJ1(t,i)-
RINJmu)/RINJstdv;(CONC(t,i)-CONCmu)/CONCstdv];
                    end
        end
    end
end


xorg=x;
Y=[x;NPVinit];



Yorg=Y;
```

```
for t=1:Ne %To generate vector of ones used to remove the mean from the realizations

    h(t)=1;
end

for t=1:(Ny) %To generate vector that contains the means of each row of the realizations
    Yavg(t)=mean(Y(t,:));

end

B=Y-Yavg'*h;
Cy=(1/(Ne-1))*B*B';

%%
% NPVn=Yorg(Ny,1);
NPVn=mean(Yorg(Ny,:));
%%
NPVn1=2*abs(NPVn);
n=1;
it=n;
ALPHAstore={};
NPValphastore={};
esstore={};

while NPVn1>=NPVn
disp(['Optimization Iteration #',num2str(n)]);
    %Realization optimizations



    [Y,ALPHAstore,NPValphastore,esstore] =
realization_opt(ALPHAinit,DALPHA,Cy,Y,i,M,Ny,Nt,h,Ne,n,TSTEP,DT,grp,CP1min,CP2min,CP3min,
CP4min,RINJ1min,CONCmin,CP1max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,PCON,op_p,wp_p,winj_
p,surf_p,r,ALPHAstore,NPValphastore,ALPHAlow,ALPHAhigh,maxit,es,esstore,TRANSFORM);







    if n>1
        NPVn=NPVn1;

    end

    Yit(:,:,n)=Y;

    for i=1:Ne
        NPVreal(n,i)=Yit(Ny,i,n);

    end

    NPVmeanit(n)=mean(Y(Ny,:));
    %%
%      NPVn1=Y(Ny,1);
     NPVn1=mean(Y(Ny,:));
    %%
    if n==1
    ecop(n)=100*((mean(NPVreal(n,:))-mean(NPVinit(i)))/mean(NPVinit(i)));
    end



    if n>1
    ecop(n)=100*((NPVmeanit(n)-NPVmeanit(n-1))/NPVmeanit(n-1));
    end
```

```
    if NPVn1>=NPVn

        close all;
        figure;
        plot([0,it],[mean(NPVinit),NPVmeanit(1:n)],'-k');
        xlabel('iteration');
        ylabel('Average Net Present Value, $');
        title('Iterations versus Average Net Present Value');

        for i=1:Graph
            figure;
            plot([0,it],[NPVinit(i);NPVreal(1:n,i)],'-k');
            xlabel('iteration');
            ylabel('Net Present Value, $');
            title(['Iterations versus NPV for Realization #', num2str(i)]);
        end

        for t=1:(Ny) %To generate vector that contains the means of each row of the
realizations
            Yavg(t)=mean(Y(t,:));

        end

        B=Y-Yavg'*h;
        Cy=(1/(Ne-1))*B*B';
        n=n+1;
        it(n)=n;
%       else break
    end

end %End of "while" loop for n iterations
%%
MeanNPVinit=mean(Yorg(Ny,:));
MeanNPVopt=mean(Yit(Ny,:,n-1));
MeanNPVincrease=100*(MeanNPVopt-MeanNPVinit)/MeanNPVinit;

for i=1:Ne
NPVincrease(i)=100*(NPVreal(n-1,i)-NPVinit(i))/(NPVinit(i));

end

xoptimal=Yit(1:Nt,:,n-1);

FWCTinitf={};
WCT1initf={};
WCT2initf={};
WCT3initf={};
WCT4initf={};
FOPTinitf={};
FWPTinitf={};
FOSATinitf={};
FPRinitf={};
WBHP1initf={};
WBHP2initf={};
WBHP3initf={};
WBHP4initf={};
LP1initf={};
LP2initf={};
LP3initf={};
LP4initf={};
RINJinitf={};
CONCinitf={};
POREINJinitf={};
SURFPRODTinitf={};
```

```
TIME_YEARSinitf={};

FWCToptf={};
WCT1optf={};
WCT2optf={};
WCT3optf={};
WCT4optf={};
FOPToptf={};
FWPToptf={};
FOSAToptf={};
FPRoptf={};
WBHP1optf={};
WBHP2optf={};
WBHP3optf={};
WBHP4optf={};
LP1optf={};
LP2optf={};
LP3optf={};
LP4optf={};
RINJoptf={};
CONCoptf={};
POREINJoptf={};
SURFPRODToptf={};
TIME_YEARSoptf={};

[xorgt]=xt;
[xoptimalt]=transform_xNe(xoptimal,TSTEP,grp,Ne,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONC
min,CP1max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM);



for i=1:Ne
[NPVinitf(i),
Poilinitf(i),FWCTinit,FOPTinit,FWPTinit,FOSATinit,FPRinit,TIME_YEARSinit,WCT1init,WCT2ini
t,WCT3init,WCT4init,WBHP_P1init,WBHP_P2init,WBHP_P3init,WBHP_P4init,WLPR_P1init,WLPR_P2in
it,WLPR_P3init,WLPR_P4init,FWIRinit,CONCinit,POREINJinit,SURFPRODTinit]=surfblackbox3(TST
EP,DT,grp,i,xorgt,PCON,op_p,wp_p,winj_p,surf_p,r,PVTOTAL);
FWCTinitf(i)={FWCTinit};
WCT1initf(i)={WCT1init};
WCT2initf(i)={WCT2init};
WCT3initf(i)={WCT3init};
WCT4initf(i)={WCT4init};
FOPTinitf(i)={FOPTinit};
FWPTinitf(i)={FWPTinit};
FOSATinitf(i)={FOSATinit};
FPRinitf(i)={FPRinit};
WBHP1initf(i)={WBHP_P1init};
WBHP2initf(i)={WBHP_P2init};
WBHP3initf(i)={WBHP_P3init};
WBHP4initf(i)={WBHP_P4init};
LP1initf(i)={WLPR_P1init};
LP2initf(i)={WLPR_P2init};
LP3initf(i)={WLPR_P3init};
LP4initf(i)={WLPR_P4init};
RINJinitf(i)={FWIRinit};
CONCinitf(i)={CONCinit};
POREINJinitf(i)={POREINJinit};
SURFPRODTinitf(i)={SURFPRODTinit};
TIME_YEARSinitf(i)={TIME_YEARSinit};

RFinit(i)= 100*Poilinitf(i)/OIP;

[NPVoptf(i),
Poiloptf(i),FWCTopt,FOPTopt,FWPTopt,FOSATopt,FPRopt,TIME_YEARSopt,WCT1opt,WCT2opt,WCT3opt
,WCT4opt,WBHP_P1opt,WBHP_P2opt,WBHP_P3opt,WBHP_P4opt,WLPR_P1opt,WLPR_P2opt,WLPR_P3opt,WLP
R_P4opt,FWIRopt,CONCopt,POREINJopt,SURFPRODTopt]=surfblackbox3(TSTEP,DT,grp,i,xoptimalt,P
CON,op_p,wp_p,winj_p,surf_p,r,PVTOTAL); %n-1 possibly for optimal
```

```
FWCToptf(i)={FWCTopt};
WCT1optf(i)={WCT1opt};
WCT2optf(i)={WCT2opt};
WCT3optf(i)={WCT3opt};
WCT4optf(i)={WCT4opt};
FOPToptf(i)={FOPTopt};
FWPToptf(i)={FWPTopt};
FOSAToptf(i)={FOSATopt};
FPRoptf(i)={FPRopt};
WBHP1optf(i)={WBHP_P1opt};
WBHP2optf(i)={WBHP_P2opt};
WBHP3optf(i)={WBHP_P3opt};
WBHP4optf(i)={WBHP_P4opt};
LP1optf(i)={WLPR_P1opt};
LP2optf(i)={WLPR_P2opt};
LP3optf(i)={WLPR_P3opt};
LP4optf(i)={WLPR_P4opt};
RINJoptf(i)={FWIRopt};
CONCoptf(i)={CONCopt};
POREINJoptf(i)={POREINJopt};
SURFPRODToptf(i)={SURFPRODTopt};
TIME_YEARSoptf(i)={TIME_YEARSopt};

RFopt(i) = 100*Poiloptf(i)/OIP;
clear FWCTopt FOPTopt FWPTopt FOSATopt FPRopt TIME_YEARSopt WCT1opt WCT2opt WCT3opt
WCT4opt WBHP_P1opt WBHP_P2opt WBHP_P3opt WBHP_P4opt WLPR_P1opt WLPR_P2opt WLPR_P3opt
WLPR_P4opt FWIRopt CONCopt FWCTinit FOPTinit FWPTinit FOSATinit FPRinit TIME_YEARSinit
WCT1init WCT2init WCT3init WCT4init WBHP_P1init WBHP_P2init WBHP_P3init WBHP_P4init
WLPR_P1init WLPR_P2init WLPR_P3init WLPR_P4init FWIRinit CONCinit POREINJopt SURFPRODTopt
POREINJinit SURFPRODTinit;
end


for i=1:Ne
Poilincrease(i)=100*(Poiloptf(i)-Poilinitf(i))/(Poilinitf(i));
RFincrease(i)=100*(RFopt(i)-RFinit(i))/(RFinit(i));

end




figure;
bar(Poilinitf);
xlabel('Realization');
ylabel('Cumulative Oil, STB');
title('Inital Cumulative Oil');

figure;
bar(Poiloptf);
xlabel('Realization');
ylabel('Cumulative Oil, STB');
title('Optimized Cumulative Oil');

figure;
bar(Poilincrease);
xlabel('Realization');
ylabel('Increase Cumulative Oil, %');
title('Increase Cumulative Oil');


%%
figure;
bar(NPVinit);
xlabel('Realization');
ylabel('NPV, $');
title('Original NPV distribution');
```

```
figure;
bar(NPVreal(n-1,:));
xlabel('Realization');
ylabel('NPV, $');
title('Optimized NPV distribution');

figure;
bar(NPVincrease);
xlabel('Realization');
ylabel('Increase in NPV, %');
title('Increase in NPV');

%%

figure;
bar(RFinit);
xlabel('Realization');
ylabel('Inital Recovery Factor, %');
title('Inital Recovery Factor');

figure;
bar(RFopt);
xlabel('Realization');
ylabel('Optimized Recovery Factor, %');
title('Optimized Recovery Factor');

figure;
bar(RFincrease);
xlabel('Realization');
ylabel('Recovery Factor Increase, %');
title('Recovery Factor Increase');

%% Optimized Controls Graphs

for i=1:Graph

figure;
for t=1:TSTEP
    C1plot(t)=xoptimalt(grp(t)-5,i); %change
    C2plot(t)=xoptimalt(grp(t)-4,i); %change
    C3plot(t)=xoptimalt(grp(t)-3,i); %change
    C4plot(t)=xoptimalt(grp(t)-2,i); %change
    C5plot(t)=xoptimalt(grp(t)-1,i); %change
    C6plot(t)=xoptimalt(grp(t),i); %change
end

if PCON==0
plot(period,C1plot,'or');
hold on
plot(period,C2plot,'xg');
hold on
plot(period,C3plot,'sk');
hold on
plot(period,C4plot,'*m');
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized BHP Profile for Realization#',num2str(i)]);
legend('P1','P2','P3','P4')
end

if PCON==1
plot(period,C1plot,'or');
hold on
plot(period,C2plot,'xg');
hold on
plot(period,C3plot,'sk');
```

```
hold on
plot(period,C4plot,'*m');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Oil Flowrate Profile for Realization#',num2str(i)]);
legend('P1','P2','P3','P4')
end

if PCON==2
plot(period,C1plot,'or');
hold on
plot(period,C2plot,'xg');
hold on
plot(period,C3plot,'sk');
hold on
plot(period,C4plot,'*m');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Total Flowrate Profile for Realization#',num2str(i)]);
legend('P1','P2','P3','P4')
end

figure;
plot(period,C5plot,'xk');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Injection Rate Profile for Realization#',num2str(i)]);
legend('INJ1')

figure;
plot(period,C6plot,'xk');
xlabel('Time, Months');
ylabel('LB/STB');
title(['Optimized Concentration of Surfactant Injected Profile for
Realization#',num2str(i)]);



figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FWCToptf(i)));
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WCT1optf(i)),'-or');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WCT2optf(i)),'-xg');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WCT3optf(i)),'-sk');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WCT4optf(i)),'-*m');
xlabel('Time, Months');
ylabel('fraction');
legend('Field','P1','P2','P3','P4');
title(['Optimized Watercut Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FOPToptf(i)));
xlabel('Time, Months');
ylabel('STB');
title(['Optimized Cumulative Oil Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FWPToptf(i)));
xlabel('Time, Months');
ylabel('STB');
title(['Optimized Cumulative Water Production Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FOSAToptf(i)));
```

```
xlabel('Time, Months');
ylabel('fraction');
title(['Optimized Oil Saturation Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FPRoptf(i)));
xlabel('Time, Months');
ylabel('psia');
title(['Optimized Field Pressure Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP1optf(i)),'-or');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP2optf(i)),'-xg');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP3optf(i)),'-sk');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP4optf(i)),'-*m');
xlabel('Time, Months');
ylabel('PSIA');
legend('P1','P2','P3','P4');
title(['Optimized Bottom Hole Pressure Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP1optf(i)),'-or');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP2optf(i)),'-xg');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP3optf(i)),'-sk');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP4optf(i)),'-*m');
xlabel('Time, Months');
ylabel('STB/DAY');
legend('P1','P2','P3','P4');
title(['Optimized Liquid Production Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(RINJoptf(i)));
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Injection Rate Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(CONCoptf(i)));
xlabel('Time, Months');
ylabel('LB/STB');
title(['Optimized Injected Surfactant Concentration Profile for
Realization#',num2str(i)]);
end

%% Original Controls Graphs

for i=1:Graph

figure;
for t=1:TSTEP
    C1orgplot(t)=xorgt(grp(t)-5,i); %change
    C2orgplot(t)=xorgt(grp(t)-4,i); %change
    C3orgplot(t)=xorgt(grp(t)-3,i); %change
    C4orgplot(t)=xorgt(grp(t)-2,i); %change
    C5orgplot(t)=xorgt(grp(t)-1,i); %change
    C6orgplot(t)=xorgt(grp(t),i); %change
end
if PCON==0
plot(period,C1orgplot,'or');
hold on
plot(period,C2orgplot,'xg');
```

```
hold on
plot(period,C3orgplot,'sk');
hold on
plot(period,C4orgplot,'*m');
xlabel('Time, Months');
ylabel('PSIA');
title(['Original BHP Profile for Realization#',num2str(i)]);
legend('P1','P2','P3','P4')
end

if PCON==1
plot(period,C1orgplot,'or');
hold on
plot(period,C2orgplot,'xg');
hold on
plot(period,C3orgplot,'sk');
hold on
plot(period,C4orgplot,'*m');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Oil Flowrate Profile for Realization#',num2str(i)]);
legend('P1','P2','P3','P4')
end

if PCON==2
plot(period,C1orgplot,'or');
hold on
plot(period,C2orgplot,'xg');
hold on
plot(period,C3orgplot,'sk');
hold on
plot(period,C4orgplot,'*m');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Total Flowrate Profile for Realization#',num2str(i)]);
legend('P1','P2','P3','P4')
end

figure;
plot(period,C5orgplot,'xk');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Injection Rate Profile for Realization#',num2str(i)]);
legend('INJ1')

figure;
plot(period,C6orgplot,'xk');
xlabel('Time, Months');
ylabel('lb/STB');
title(['Original Concentration of Surfactant Injected Profile for
Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FWCTinitf(i)));
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WCT1initf(i)),'-or');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WCT2initf(i)),'-xg');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WCT3initf(i)),'-sk');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WCT4initf(i)),'-*m');
legend('Field','P1','P2','P3','P4');
xlabel('Time, Months');
ylabel('fraction');
title(['Original Watercut Profile for Realization#',num2str(i)]);
```

```
figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FOPTinitf(i)));
xlabel('Time, Months');
ylabel('STB');
title(['Original Cumulative Oil Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FWPTinitf(i)));
xlabel('Time, Months');
ylabel('STB');
title(['Original Cumulative Water Production Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FOSATinitf(i)));
xlabel('Time, Months');
ylabel('fraction');
title(['Original Oil Saturation Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FPRinitf(i)));
xlabel('Time, Months');
ylabel('psia');
title(['Original Field Pressure Profile for Realization#',num2str(i)]);


figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP1initf(i)),'-or');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP2initf(i)),'-xg');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP3initf(i)),'-sk');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP4initf(i)),'-*m');
xlabel('Time, Months');
ylabel('PSIA');
legend('P1','P2','P3','P4');
title(['Original Bottom Hole Pressure Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP1initf(i)),'-or');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP2initf(i)),'-xg');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP3initf(i)),'-sk');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP4initf(i)),'-*m');
xlabel('Time, Months');
ylabel('STB/DAY');
legend('P1','P2','P3','P4');
title(['Original Liquid Production Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(RINJinitf(i)));
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Injection Rate Profile for Realization#',num2str(i)]);

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(CONCinitf(i)));
xlabel('Time, Months');
ylabel('LB/STB');
title(['Original Injected Surfactant Concentration Profile for
Realization#',num2str(i)]);
end

%% Weighting factor plot
figure;
```

```
for k=1:n
    semilogx(cell2mat(ALPHAstore(k)),cell2mat(NPValphastore(k)),'-d');
%     semilogx(cell2mat(ALPHAstore(k,1)),cell2mat(NPValphastore(k,1)),'-d');
%     loglog(cell2mat(ALPHAstore(k,1)),cell2mat(NPValphastore(k,1)),'-d');
    if k<n
    hold on
    end

end
xlabel('ALPHA');
ylabel('NPV, $');
title('NPV versus ALPHA');

figure;
for k=1:n
    plot(1:length(cell2mat(esstore(k))),cell2mat(esstore(k)),'-d');

    if k<n
    hold on
    end

end
xlabel('ITERATIONS');
ylabel('ERROR, %');
title('ERROR in ALPHA versus ITERATIONS');
%% Probabilistic Plots


for vperm=1:NR
    simrn=vperm;
    [Perm(vperm),por(vperm)]=perm_por2(PERMALL(:,vperm),PORO_MEAN);
    for vop=1:NR
         for vwp=1:NR
             for vwi=1:NR
                     for vsurf=1:NR
                        i=1;
                        disp(['Permeability #',num2str(vperm),'Oil Price
#',num2str(vop),'Water Treatment Cost #',num2str(vwp),'Water Injection Cost
#',num2str(vwi),'Surfactant Cost #',num2str(vsurf)]);

                        if simrn==vperm
                            [NPVhist(vsurf,vwi,vwp,vop,vperm),
Poilhist(vsurf,vwi,vwp,vop,vperm)]=surfblackbox1(TSTEP,DT,grp,i,xoptimalt,PCON,op_preal(v
op),wp_p,winj_p,surf_p,r);
                            simrn=0;
                        end

                        if simrn==0
                            [NPVhist(vsurf,vwi,vwp,vop,vperm),
Poilhist(vsurf,vwi,vwp,vop,vperm)]=surfblackbox1econ(i,op_preal(vop),wp_preal(vwp),winj_p
real(vwi),surf_preal(vsurf),r);
                        end

                     end
             end
         end
     end
end

NPVprob=reshape(NPVhist,NR^5,1);
Poilprob=reshape(Poilhist,NR^5,1);
figure;
ecdf(NPVprob);
xlabel('NPV, $');
ylabel('Probability');
title(['Probabilistic NPV for Realization', num2str(1)]);
```

```
figure;
ecdf(Poilprob);
xlabel('Cumulative Production, STB');
ylabel('Probability');
title(['Probabilistic Cumulative Production for Realization',num2str(1)]);
%%
RUNTIME=toc;
if PCON==0
save (['Opt Run Area_320_surf_flood20yrs#',num2str(rNe),' with ',num2str(Ne),'# Realz
with BHP control',' with mean',num2str(NR),'#Prob Realz Alphaconv is ',num2str(es),' Sw
is ',num2str(S_w),' TRANSFORM is ',num2str(TRANSFORM)]);
end

if PCON==1
save (['Opt Run Area_320_surf_flood20yrs#',num2str(rNe),' with ',num2str(Ne),'# Realz
with Oil Flowrate control',' with mean',num2str(NR),'#Prob Realz Alphaconv is
',num2str(es),' Sw is ',num2str(S_w),' TRANSFORM is ',num2str(TRANSFORM)]);
end

if PCON==2
save (['Opt Run Area_320_surf_flood20yrs#',num2str(rNe),' with ',num2str(Ne),'# Realz
with Liquid Flowrate control',' with mean',num2str(NR),'#Prob Realz Alphaconv is
',num2str(es),' Sw is ',num2str(S_w),' TRANSFORM is ',num2str(TRANSFORM)]);
end

end
```

SURFACTANT SIMULATION PROGRAM 1

This program writes the controls to Eclipse for all user given time intervals, runs Eclipse, reads the RSM file, and calculates the cumulative oil and NPV.

surfblackbox1.m

```
function [NetPV, CUMOIL]=surfblackbox1(TSTEP,DT,grp,i,x,PCON,op_p,wp_p,winj_p,surf_p,r)
disp(['For Realization #',num2str(i)]);
%%
for t=1:TSTEP



    if PCON == 0
    fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid, 'WCONPROD');
    fprintf(fid, '\n');
    fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-5,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-4,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-3,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');
```

```
    fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-2,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fclose(fid);
    end

    if PCON == 1

    fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid, 'WCONPROD');
    fprintf(fid, '\n');
    fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-5,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-4,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-3,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-2,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fclose(fid);
    end

    if PCON == 2
    fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid, 'WCONPROD');
    fprintf(fid, '\n');
    fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-5,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-4,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-3,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-2,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fclose(fid);
    end

    fid1 = fopen(['INJ_CONTROLS',num2str(t),'.IN'], 'w+t');
```

```
    fprintf(fid1, 'WCONINJE');
    fprintf(fid1, '\n');
    fprintf(fid1, ['''INJ', num2str(1), ''' ', '''WATER'' ''OPEN'' ''RATE'' ']);
    fprintf(fid1, num2str(x(grp(t)-1,i)));
    fprintf(fid1, ' 5*/');
    fprintf(fid1, '\n');
    fprintf(fid1, '/');
    fprintf(fid1, '\n');
    fprintf(fid1, '\r');
    fclose(fid1);

    fid2 = fopen(['CONC_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid2, 'WSURFACT');
    fprintf(fid2, '\n');
    fprintf(fid2, ['''INJ', num2str(1), ''' ']);
    fprintf(fid2, num2str(x(grp(t),i)));
    fprintf(fid2, '/');
    fprintf(fid2, '\n');
    fprintf(fid2, '/');
    fclose(fid2);

end
    fid3 = fopen('TIME_CONTROLS.IN', 'wt');
    fprintf(fid3, 'TSTEP');
    fprintf(fid3, '\n');
    fprintf(fid3, [num2str(1), '*']);
    fprintf(fid3, num2str(DT));
    fprintf(fid3, '/');
    fprintf(fid3, '\n');
    fclose(fid3);

    fid4 = fopen('ALL.IN', 'w+t');
    for t=1:TSTEP
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, '''Wflood_sch.INC''');
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''PRO_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''INJ_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''CONC_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''TIME_CONTROLS.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, '\n');
    end
    fclose(fid4);



    [STAT RESULTS] = dos('$eclipse SURFACT'); clear RESULTS
```

```
%%
if STAT==1
%%  read labels data from SURFACT.RSM
[labels1,TIME_DAYS1,OBS] = readColData('SURFACT.RSM',10,6);
[labels2,TIME_DAYS2,OBS2] = readColData('SURFACT.RSM',10,length(TIME_DAYS1)+13);
[labels3,TIME_DAYS3,OBS3] =
readColData('SURFACT.RSM',10,length(TIME_DAYS2)+length(TIME_DAYS1)+20);
[labels4,TIME_DAYS4,OBS4] =
readColData('SURFACT.RSM',10,length(TIME_DAYS3)+length(TIME_DAYS2)+length(TIME_DAYS1)+27)
;
fclose('all');
% pause(2);
LV=length(TIME_DAYS1);
TIME_YEARS=OBS(:,1);
WBHP_P1=OBS(:,2);
WBHP_P2=OBS(:,3);
WBHP_P3=OBS(:,4);
WBHP_P4=OBS(:,5);
WBHP_INJ1=OBS(:,6);
FWIR=OBS(:,7);
FOPR=OBS(:,8);
FPR=OBS(:,9);



FTPRSUR=OBS2(:,1);
FOE=OBS2(:,2);


FTIRSUR=OBS2(:,3);
FOSAT=OBS2(:,4);


WTPRSUR=OBS2(:,5);
FWPR=OBS2(:,6);



FGPR=OBS2(:,7);

FWCT=OBS2(:,8);
FWSAT=OBS2(:,9);


FWIR=OBS3(:,1);
WWCT_P1=OBS3(:,2);
WWCT_P2=OBS3(:,3);
WWCT_P3=OBS3(:,4);
WWCT_P4=OBS3(:,5);
WWCT_INJ1=OBS3(:,6);
WOPT_P1=OBS3(:,7);
WOPT_P2=OBS3(:,8);
WOPT_P3=OBS3(:,9);
WOPT_P4=OBS4(:,1);
WOPT_INJ1=OBS4(:,2);
WOPR_P1=OBS4(:,3);
WOPR_P2=OBS4(:,4);
WOPR_P3=OBS4(:,5);
WOPR_P4=OBS4(:,6);
WOPR_INJ1=OBS4(:,7);
%%


    for n=1:LV;
        if n==1
        FOPT(n)=FOPR(n);
```

```
        FTITSUR(n)=FTIRSUR(n);
        FWIT(n)=FWIR(n);
        FTPTSUR(n)=FTPRSUR(n);
        FWPT(n)=FWPR(n);

        OPR(n)=FOPT(n);
        WPR(n)=FWPT(n);
        WINJ(n)=FWIT(n);
        SINJ(n)=FTITSUR(n);
        end

        if n>1
        FOPT(n)=FOPT(n-1)+FOPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTITSUR(n)=FTITSUR(n-1)+FTIRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWIT(n)=FWIT(n-1)+FWIR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTPTSUR(n)=FTPTSUR(n-1)+FTPRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWPT(n)=FWPT(n-1)+FWPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));

        OPR(n)=FOPT(n)-FOPT(n-1);
        WPR(n)=FWPT(n)-FWPT(n-1);
        WINJ(n)=FWIT(n)-FWIT(n-1);
        SINJ(n)=FTITSUR(n)-FTITSUR(n-1);
        end

end

for n=1:LV;
    Ct(n)=OPR(n)*op_p-WPR(n)*wp_p-WINJ(n)*winj_p-SINJ(n)*surf_p;

    Inner(n)=Ct(n)/((1+r/(100*365))^TIME_DAYS1(n));
end





clear STAT;
end

NetPV=sum(Inner);
CUMOIL=FOPT(LV);
clear OPR WPR WINJ SINJ Ct Inner NPV STAT;

end
```

SURFACTANT SIMULATION PROGRAM 2

This program writes the controls to Eclipse for select time intervals, runs Eclipse, reads the RSM file, and outputs reservoir properties from the simulation run.

surfblackbox2.m

```
function
[FLRp,PVupdate,VOIDp,FLTp,WCT,FTOTSUR,SINJrate,kro1,kro2,kro3,kro4,krofield,krw1,krw2,krw
3,krw4,Swfield,Pi_field,krwfield,LPR_P1,LPR_P2,LPR_P3,LPR_P4,DRWD1,DRWD2,DRWD3,DRWD4,WCT1
,WCT2,WCT3,WCT4,comp]=surfblackbox2(TSTEP,DT,grp,i,x,PCON,op_p,wp_p,winj_p,surf_p,r,OIP1,
OIP2,OIP3,OIP4,FVF_o,por1,por2,por3,por4,pay,Area1,Area2,Area3,Area4,Sw,kro,krw,PV)
disp(['Property Estimation for Realization #',num2str(i),' timstep #',num2str(TSTEP)]);
%%
for t=1:TSTEP
```

```
if PCON == 0
fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
fprintf(fid, 'WCONPROD');
fprintf(fid, '\n');
fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
fprintf(fid, num2str(x(grp(t)-5,i)));
fprintf(fid, '/');
fprintf(fid, '\n');

fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
fprintf(fid, num2str(x(grp(t)-4,i)));
fprintf(fid, '/');
fprintf(fid, '\n');

fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
fprintf(fid, num2str(x(grp(t)-3,i)));
fprintf(fid, '/');
fprintf(fid, '\n');

fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
fprintf(fid, num2str(x(grp(t)-2,i)));
fprintf(fid, '/');
fprintf(fid, '\n');
fprintf(fid, '/');
fprintf(fid, '\n');
fclose(fid);
end

if PCON == 1

fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
fprintf(fid, 'WCONPROD');
fprintf(fid, '\n');
fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''ORAT''', ' ']);
fprintf(fid, num2str(x(grp(t)-5,i)));
fprintf(fid, '/');
fprintf(fid, '\n');

fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''ORAT''', ' ']);
fprintf(fid, num2str(x(grp(t)-4,i)));
fprintf(fid, '/');
fprintf(fid, '\n');

fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''ORAT''', ' ']);
fprintf(fid, num2str(x(grp(t)-3,i)));
fprintf(fid, '/');
fprintf(fid, '\n');

fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''ORAT''', ' ']);
fprintf(fid, num2str(x(grp(t)-2,i)));
fprintf(fid, '/');
fprintf(fid, '\n');
fprintf(fid, '/');
fprintf(fid, '\n');
fclose(fid);
end

if PCON == 2
fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
fprintf(fid, 'WCONPROD');
fprintf(fid, '\n');
fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
fprintf(fid, num2str(x(grp(t)-5,i)));
fprintf(fid, '/');
```

```
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-4,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-3,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-2,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fclose(fid);
    end

    fid1 = fopen(['INJ_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid1, 'WCONINJE');
    fprintf(fid1, '\n');
    fprintf(fid1, ['''INJ', num2str(1), ''' ', '''WATER'' ''OPEN'' ''RATE'' ']);
    fprintf(fid1, num2str(x(grp(t)-1,i)));
    fprintf(fid1, ' 5*/');
    fprintf(fid1, '\n');
    fprintf(fid1, '/');
    fprintf(fid1, '\n');
    fprintf(fid1, '\r');
    fclose(fid1);

    fid2 = fopen(['CONC_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid2, 'WSURFACT');
    fprintf(fid2, '\n');
    fprintf(fid2, ['''INJ', num2str(1), ''' ']);
    fprintf(fid2, num2str(x(grp(t),i)));
    fprintf(fid2, '/');
    fprintf(fid2, '\n');
    fprintf(fid2, '/');
    fclose(fid2);

end
    fid3 = fopen('TIME_CONTROLS.IN', 'wt');
    fprintf(fid3, 'TSTEP');
    fprintf(fid3, '\n');
    fprintf(fid3, [num2str(1), '*']);
    fprintf(fid3, num2str(DT));
    fprintf(fid3, '/');
    fprintf(fid3, '\n');
    fclose(fid3);

    fid4 = fopen('ALL.IN', 'w+t');
    for t=1:TSTEP
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, '''Wflood_sch.INC''');
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''PRO_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
```

```
    fprintf(fid4, ['''INJ_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''CONC_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''TIME_CONTROLS.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, '\n');
    end
    fclose(fid4);




    [STAT RESULTS] = dos('$eclipse SURFACT'); clear RESULTS


%%
if STAT==1
%%  read labels data from SURFACT.RSM
[labels1,TIME_DAYS1,OBS] = readColData('SURFACT.RSM',10,6);
[labels2,TIME_DAYS2,OBS2] = readColData('SURFACT.RSM',10,length(TIME_DAYS1)+13);
[labels3,TIME_DAYS3,OBS3] =
readColData('SURFACT.RSM',10,length(TIME_DAYS2)+length(TIME_DAYS1)+20);
[labels4,TIME_DAYS4,OBS4] =
readColData('SURFACT.RSM',10,length(TIME_DAYS3)+length(TIME_DAYS2)+length(TIME_DAYS1)+27)
;
[labels5,TIME_DAYS5,OBS5] =
readColData('SURFACT.RSM',4,length(TIME_DAYS4)+length(TIME_DAYS3)+length(TIME_DAYS2)+leng
th(TIME_DAYS1)+34);
fclose('all');
% pause(2);
LV=length(TIME_DAYS1);
TIME_YEARS=OBS(:,1);
WBHP_P1=OBS(:,2);
WBHP_P2=OBS(:,3);
WBHP_P3=OBS(:,4);
WBHP_P4=OBS(:,5);
WBHP_INJ1=OBS(:,6);
FWIR=OBS(:,7);
FOPR=OBS(:,8);
FPR=OBS(:,9);



FTPRSUR=OBS2(:,1);
FOE=OBS2(:,2);



FTIRSUR=OBS2(:,3);
FOSAT=OBS2(:,4);


WTPRSUR=OBS2(:,5);
FWPR=OBS2(:,6);



FGPR=OBS2(:,7);
```

```
FWCT=OBS2(:,8);
FWSAT=OBS2(:,9);

FWIR=OBS3(:,1);
WWCT_P1=OBS3(:,2);
WWCT_P2=OBS3(:,3);
WWCT_P3=OBS3(:,4);
WWCT_P4=OBS3(:,5);
WWCT_INJ1=OBS3(:,6);
WOPT_P1=OBS3(:,7);
WOPT_P2=OBS3(:,8);
WOPT_P3=OBS3(:,9);
WOPT_P4=OBS4(:,1);
WOPT_INJ1=OBS4(:,2);
WOPR_P1=OBS4(:,3);
WOPR_P2=OBS4(:,4);
WOPR_P3=OBS4(:,5);
WOPR_P4=OBS4(:,6);
WOPR_INJ1=OBS4(:,7);
WLPR_P1=OBS4(:,8);
WLPR_P2=OBS4(:,9);
WLPR_P3=OBS5(:,1);
WLPR_P4=OBS5(:,2);
%%


    for n=1:LV;
        if n==1
        FOPT(n)=FOPR(n);
        FTITSUR(n)=FTIRSUR(n);
        FWIT(n)=FWIR(n);
        FTPTSUR(n)=FTPRSUR(n);
        FWPT(n)=FWPR(n);
        FLT(n)=FOPT(n)+FWPT(n);
        VOID(n)=FWIT(n)-FLT(n);

        OPR(n)=FOPT(n);
        WPR(n)=FWPT(n);
        WINJ(n)=FWIT(n);
        SINJ(n)=FTITSUR(n);
        end

        if n>1
        FOPT(n)=FOPT(n-1)+FOPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTITSUR(n)=FTITSUR(n-1)+FTIRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWIT(n)=FWIT(n-1)+FWIR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTPTSUR(n)=FTPTSUR(n-1)+FTPRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWPT(n)=FWPT(n-1)+FWPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FLT(n)=FOPT(n)+FWPT(n);
        VOID(n)=FWIT(n)-FLT(n);

        OPR(n)=FOPT(n)-FOPT(n-1);
        WPR(n)=FWPT(n)-FWPT(n-1);
        WINJ(n)=FWIT(n)-FWIT(n-1);
        SINJ(n)=FTITSUR(n)-FTITSUR(n-1);
        end

end

for n=1:LV;
    Ct(n)=OPR(n)*op_p-WPR(n)*wp_p-WINJ(n)*winj_p-SINJ(n)*surf_p;

    Inner(n)=Ct(n)/((1+r/(100*365))^TIME_DAYS1(n));
end


clear STAT;
```

```
end

Sw1=1-((OIP1-WOPT_P1(LV))*FVF_o/(7758*por1*pay*Area1));
Sw2=1-((OIP2-WOPT_P2(LV))*FVF_o/(7758*por2*pay*Area2));
Sw3=1-((OIP3-WOPT_P3(LV))*FVF_o/(7758*por3*pay*Area3));
Sw4=1-((OIP4-WOPT_P4(LV))*FVF_o/(7758*por4*pay*Area4));
kro1=interp1([0,Sw,1],[1,kro,0],Sw1);
kro2=interp1([0,Sw,1],[1,kro,0],Sw2);
kro3=interp1([0,Sw,1],[1,kro,0],Sw3);
kro4=interp1([0,Sw,1],[1,kro,0],Sw4);



krw1=interp1([0,Sw,1],[0,krw,1],Sw1);
krw2=interp1([0,Sw,1],[0,krw,1],Sw2);
krw3=interp1([0,Sw,1],[0,krw,1],Sw3);
krw4=interp1([0,Sw,1],[0,krw,1],Sw4);

Swfield=FWSAT(LV);
krofield=interp1([0,Sw,1],[1,kro,0],Swfield);
krwfield=interp1([0,Sw,1],[0,krw,1],Swfield);


WCT=FWCT(LV);
FTOTSUR=FTPTSUR(LV);
SINJrate=FTPRSUR(LV);

Pi_field=FPR(LV);
FLTp=FLT(LV);
FLRp=OPR(LV)+WPR(LV);
VOIDp=VOID(LV);
PVupdate=PV-(FLTp-FWIT(LV));

PVupdatehigh=PV-(FLT(LV)-FWIT(LV));
PVupdatelow=PV-(FLT(LV-1)-FWIT(LV-1));
AVPV=(PVupdatehigh+PVupdatelow)/2;
comp=-(1/AVPV)*(PVupdatehigh-PVupdatelow)/(FPR(LV-1)-FPR(LV));

LPR_P1=WLPR_P1(LV);
LPR_P2=WLPR_P2(LV);
LPR_P3=WLPR_P3(LV);
LPR_P4=WLPR_P4(LV);
DRWD1=FPR(LV)-WBHP_P1(LV);
DRWD2=FPR(LV)-WBHP_P2(LV);
DRWD3=FPR(LV)-WBHP_P3(LV);
DRWD4=FPR(LV)-WBHP_P4(LV);
WCT1=WWCT_P1(LV);
WCT2=WWCT_P2(LV);
WCT3=WWCT_P3(LV);
WCT4=WWCT_P4(LV);

clear OPR WPR WINJ SINJ Ct Inner NPV STAT;

end
```

SURFACTANT SIMULATION PROGRAM 3

This program writes the controls to Eclipse for all time intervals, runs Eclipse, reads the RSM file, and calculates the cumulative oil, NPV, and determines reservoir properties based on the simulation run so that graphics can be generated.

surfblackbox3.m

```
function [NetPV, CUMOIL,
FWCTt,FOPTt,FWPTt,FOSATt,FPRt,TIME_YEARSt,WWCT_P1,WWCT_P2,WWCT_P3,WWCT_P4,WBHP_P1,WBHP_P2
,WBHP_P3,WBHP_P4,WLPR_P1,WLPR_P2,WLPR_P3,WLPR_P4,FWIR,CONC,POREINJ,SURFPRODT]=surfblackbo
x3(TSTEP,DT,grp,i,x,PCON,op_p,wp_p,winj_p,surf_p,r,PVTOTAL)
disp(['For Realization #',num2str(i)]);
%%
for t=1:TSTEP
% n=t;



%     disp(['Writing for Timestep #',num2str(t)]);


    if PCON == 0
    fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid, 'WCONPROD');
    fprintf(fid, '\n');
    fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-5,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-4,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-3,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''BHP''', ' 5* ']);
    fprintf(fid, num2str(x(grp(t)-2,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fclose(fid);
    end

    if PCON == 1

    fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid, 'WCONPROD');
    fprintf(fid, '\n');
    fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-5,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-4,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-3,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''ORAT''', ' ']);
    fprintf(fid, num2str(x(grp(t)-2,i)));
    fprintf(fid, '/');
```

```
    fprintf(fid, '\n');
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fclose(fid);
    end

    if PCON == 2
    fid = fopen(['PRO_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid, 'WCONPROD');
    fprintf(fid, '\n');
    fprintf(fid, ['''P', num2str(1), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-5,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(2), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-4,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(3), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-3,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');

    fprintf(fid, ['''P', num2str(4), ''' ', '''OPEN'' ''LRAT''', ' 3* ']);
    fprintf(fid, num2str(x(grp(t)-2,i)));
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fprintf(fid, '/');
    fprintf(fid, '\n');
    fclose(fid);
    end

    fid1 = fopen(['INJ_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid1, 'WCONINJE');
    fprintf(fid1, '\n');
    fprintf(fid1, ['''INJ', num2str(1), ''' ', '''WATER'' ''OPEN'' ''RATE'' ']);
    fprintf(fid1, num2str(x(grp(t)-1,i)));
    fprintf(fid1, ' 5*/');
    fprintf(fid1, '\n');
    fprintf(fid1, '/');
    fprintf(fid1, '\n');
    fprintf(fid1, '\r');
    fclose(fid1);

    fid2 = fopen(['CONC_CONTROLS',num2str(t),'.IN'], 'w+t');
    fprintf(fid2, 'WSURFACT');
    fprintf(fid2, '\n');
    fprintf(fid2, ['''INJ', num2str(1), ''' ']);
    fprintf(fid2, num2str(x(grp(t),i)));
    fprintf(fid2, '/');
    fprintf(fid2, '\n');
    fprintf(fid2, '/');
    fclose(fid2);

end
    fid3 = fopen('TIME_CONTROLS.IN', 'wt');
    fprintf(fid3, 'TSTEP');
    fprintf(fid3, '\n');
    fprintf(fid3, [num2str(1), '*']);
    fprintf(fid3, num2str(DT));
    fprintf(fid3, '/');
    fprintf(fid3, '\n');
    fclose(fid3);

    fid4 = fopen('ALL.IN', 'w+t');
```

```
    for t=1:TSTEP
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, '''Wflood_sch.INC''');
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''PRO_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''INJ_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''CONC_CONTROLS',num2str(t),'.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, 'INCLUDE');
    fprintf(fid4, '\n');
    fprintf(fid4, ['''TIME_CONTROLS.IN''']);
    fprintf(fid4, '/');
    fprintf(fid4, '\n');
    fprintf(fid4, '\n');
    end
    fclose(fid4);



    [STAT RESULTS] = dos('$eclipse SURFACT'); clear RESULTS


%%
if STAT==1
%%  read labels data from SURFACT.RSM
[labels1,TIME_DAYS1,OBS] = readColData('SURFACT.RSM',10,6);
[labels2,TIME_DAYS2,OBS2] = readColData('SURFACT.RSM',10,length(TIME_DAYS1)+13);
[labels3,TIME_DAYS3,OBS3] =
readColData('SURFACT.RSM',10,length(TIME_DAYS2)+length(TIME_DAYS1)+20);
[labels4,TIME_DAYS4,OBS4] =
readColData('SURFACT.RSM',10,length(TIME_DAYS3)+length(TIME_DAYS2)+length(TIME_DAYS1)+27)
;
[labels5,TIME_DAYS5,OBS5] =
readColData('SURFACT.RSM',4,length(TIME_DAYS4)+length(TIME_DAYS3)+length(TIME_DAYS2)+leng
th(TIME_DAYS1)+34);
fclose('all');
% pause(2);
LV=length(TIME_DAYS1);
TIME_YEARS=OBS(:,1);
WBHP_P1=OBS(:,2); %Data is in bar to convert multiply by 14.5038 psi
WBHP_P2=OBS(:,3); %Data is in bar to convert multiply by 14.5038 psi
WBHP_P3=OBS(:,4); %Data is in bar to convert multiply by 14.5038 psi
WBHP_P4=OBS(:,5); %Data is in bar to convert multiply by 14.5038 psi
WBHP_INJ1=OBS(:,6); %Data is in bar to convert multiply by 14.5038 psi
FWIR=OBS(:,7); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
FOPR=OBS(:,8); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
FPR=OBS(:,9); %Data is in bar to convert multiply by 14.5038 psi
% FOPT=OBS(:,9); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then divide
by 5.615
```

```
FTPRSUR=OBS2(:,1); %Data is in kg to convert to lb multiply by 2.20462
FOE=OBS2(:,2); %Oil recovery factor
% FTPTSUR=OBS2(:,2); %Data is in kg to convert to lb multiply by 2.20462


FTIRSUR=OBS2(:,3); %Data is in kg to convert to lb multiply by 2.20462
FOSAT=OBS2(:,4); %Average Oil Saturation
% FTITSUR=OBS2(:,4); %Data is in kg to convert to lb multiply by 2.20462

WTPRSUR=OBS2(:,5); %Data is in kg to convert to lb multiply by 2.20462
FWPR=OBS2(:,6); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615

% FWPT=OBS2(:,7); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615

FGPR=OBS2(:,7); %Data is in SM^3 to convert to SCF multiply by 35.3147

FWCT=OBS2(:,8);
FWSAT=OBS2(:,9); %Average Oil Saturation
%FWIT=OBS2(:,9); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then divide
by 5.615

FWIR=OBS3(:,1); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WWCT_P1=OBS3(:,2);
WWCT_P2=OBS3(:,3);
WWCT_P3=OBS3(:,4);
WWCT_P4=OBS3(:,5);
WWCT_INJ1=OBS3(:,6);
WOPT_P1=OBS3(:,7); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_P2=OBS3(:,8); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_P3=OBS3(:,9); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_P4=OBS4(:,1); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_INJ1=OBS4(:,2); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P1=OBS4(:,3); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P2=OBS4(:,4); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P3=OBS4(:,5); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P4=OBS4(:,6); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_INJ1=OBS4(:,7); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3
then divide by 5.615
WLPR_P1=OBS4(:,8); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WLPR_P2=OBS4(:,9); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WLPR_P3=OBS5(:,1); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WLPR_P4=OBS5(:,2); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
%%
%Calculate NPV
% op_p=60; %Price of oil $/STB
% wp_p=20; %Price of water disposal $/STB
% winj_p=10; %Price of injected water $/STB
% surf_p=3; %Price of surfactant used $/lb
% r=10; %Discount rate in %

    for n=1:LV;
```

```
        if n==1
        FOPT(n)=FOPR(n);
        FTITSUR(n)=FTIRSUR(n);
        FWIT(n)=FWIR(n);
        FTPTSUR(n)=FTPRSUR(n);
        FWPT(n)=FWPR(n);

        SURFPRODT(n)=FTPRSUR(n);

        OPR(n)=FOPT(n);
        WPR(n)=FWPT(n);
        WINJ(n)=FWIT(n);
        SINJ(n)=FTITSUR(n);

        SURFPRODR(n)=SURFPRODT(n);
        end

        if n>1
        FOPT(n)=FOPT(n-1)+FOPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTITSUR(n)=FTITSUR(n-1)+FTIRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWIT(n)=FWIT(n-1)+FWIR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTPTSUR(n)=FTPTSUR(n-1)+FTPRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWPT(n)=FWPT(n-1)+FWPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));

        SURFPRODT(n)=SURFPRODT(n-1)+FTPRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));

        OPR(n)=FOPT(n)-FOPT(n-1);
        WPR(n)=FWPT(n)-FWPT(n-1);
        WINJ(n)=FWIT(n)-FWIT(n-1);
        SINJ(n)=FTITSUR(n)-FTITSUR(n-1);
        end
        CONC(n)=FTIRSUR(n)/FWIR(n);
        POREINJ(n)=FWIT(n)/PVTOTAL;
end

for n=1:LV;
    Ct(n)=OPR(n)*op_p-WPR(n)*wp_p-WINJ(n)*winj_p-SINJ(n)*surf_p;
    Inner(n)=Ct(n)/((1+r/(100*365))^TIME_DAYS1(n));

end

% NPV(n)=sum(Inner);



%delete SURFACT.RSM;
%clear OPR WPR WINJ SINJ Ct Inner NPV STAT;
clear STAT;
end
FOPTt=FOPT;
FWPTt=FWPT;
FWCTt=FWCT';
FOSATt=FOSAT';
FPRt=FPR';
TIME_YEARSt=TIME_YEARS';
NetPV=sum(Inner);
CUMOIL=FOPT(LV);
clear OPR WPR WINJ SINJ Ct Inner NPV STAT;
% for t=1:TSTEP
% delete (['PRO_CONTROLS',num2str(t),'.IN'])
% delete (['INJ_CONTROLS',num2str(t),'.IN'])
% delete (['CONC_CONTROLS',num2str(t),'.IN'])
% end
end
```

SURFACTANT SIMULATION PROGRAM 4

This program calculates the NPV based on the information from a run in Eclipse.

surfblackbox1econ.m

```
function [NetPV, CUMOIL]=surfblackbox1econ(i,op_p,wp_p,winj_p,surf_p,r)
disp(['For Realization #',num2str(i)]);


%%  read labels data from SURFACT.RSM
[labels1,TIME_DAYS1,OBS] = readColData('SURFACT.RSM',10,6);
[labels2,TIME_DAYS2,OBS2] = readColData('SURFACT.RSM',10,length(TIME_DAYS1)+13);
[labels3,TIME_DAYS3,OBS3] =
readColData('SURFACT.RSM',10,length(TIME_DAYS2)+length(TIME_DAYS1)+20);
[labels4,TIME_DAYS4,OBS4] =
readColData('SURFACT.RSM',10,length(TIME_DAYS3)+length(TIME_DAYS2)+length(TIME_DAYS1)+27)
;
fclose('all');
% pause(2);
LV=length(TIME_DAYS1);
TIME_YEARS=OBS(:,1);
WBHP_P1=OBS(:,2); %Data is in bar to convert multiply by 14.5038 psi
WBHP_P2=OBS(:,3); %Data is in bar to convert multiply by 14.5038 psi
WBHP_P3=OBS(:,4); %Data is in bar to convert multiply by 14.5038 psi
WBHP_P4=OBS(:,5); %Data is in bar to convert multiply by 14.5038 psi
WBHP_INJ1=OBS(:,6); %Data is in bar to convert multiply by 14.5038 psi
FWIR=OBS(:,7); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
FOPR=OBS(:,8); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
FPR=OBS(:,9); %Data is in bar to convert multiply by 14.5038 psi
% FOPT=OBS(:,9); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then divide
by 5.615


FTPRSUR=OBS2(:,1); %Data is in kg to convert to lb multiply by 2.20462
FOE=OBS2(:,2); %Oil recovery factor
% FTPTSUR=OBS2(:,2); %Data is in kg to convert to lb multiply by 2.20462


FTIRSUR=OBS2(:,3); %Data is in kg to convert to lb multiply by 2.20462
FOSAT=OBS2(:,4); %Average Oil Saturation
% FTITSUR=OBS2(:,4); %Data is in kg to convert to lb multiply by 2.20462

WTPRSUR=OBS2(:,5); %Data is in kg to convert to lb multiply by 2.20462
FWPR=OBS2(:,6); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615

% FWPT=OBS2(:,7); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615

FGPR=OBS2(:,7); %Data is in SM^3 to convert to SCF multiply by 35.3147

FWCT=OBS2(:,8);
FWSAT=OBS2(:,9); %Average Oil Saturation
%FWIT=OBS2(:,9); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then divide
by 5.615

FWIR=OBS3(:,1); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WWCT_P1=OBS3(:,2);
WWCT_P2=OBS3(:,3);
WWCT_P3=OBS3(:,4);
WWCT_P4=OBS3(:,5);
```

```
WWCT_INJ1=OBS3(:,6);
WOPT_P1=OBS3(:,7); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_P2=OBS3(:,8); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_P3=OBS3(:,9); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_P4=OBS4(:,1); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPT_INJ1=OBS4(:,2); %Data is in SM^3 to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P1=OBS4(:,3); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P2=OBS4(:,4); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P3=OBS4(:,5); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_P4=OBS4(:,6); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3 then
divide by 5.615
WOPR_INJ1=OBS4(:,7); %Data is in SM^3/Day to convert to STB multiply by 35.3147 SFT^3
then divide by 5.615
%%
%Calculate NPV
% op_p=60; %Price of oil $/STB
% wp_p=20; %Price of water disposal $/STB
% winj_p=10; %Price of injected water $/STB
% surf_p=3; %Price of surfactant used $/lb
% r=10; %Discount rate in %

    for n=1:LV;
        if n==1
        FOPT(n)=FOPR(n);
        FTITSUR(n)=FTIRSUR(n);
        FWIT(n)=FWIR(n);
        FTPTSUR(n)=FTPRSUR(n);
        FWPT(n)=FWPR(n);

        OPR(n)=FOPT(n);
        WPR(n)=FWPT(n);
        WINJ(n)=FWIT(n);
        SINJ(n)=FTITSUR(n);
        end

        if n>1
        FOPT(n)=FOPT(n-1)+FOPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTITSUR(n)=FTITSUR(n-1)+FTIRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWIT(n)=FWIT(n-1)+FWIR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FTPTSUR(n)=FTPTSUR(n-1)+FTPRSUR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));
        FWPT(n)=FWPT(n-1)+FWPR(n)*(TIME_DAYS1(n)-TIME_DAYS1(n-1));

        OPR(n)=FOPT(n)-FOPT(n-1);
        WPR(n)=FWPT(n)-FWPT(n-1);
        WINJ(n)=FWIT(n)-FWIT(n-1);
        SINJ(n)=FTITSUR(n)-FTITSUR(n-1);
        end

end

for n=1:LV;
    Ct(n)=OPR(n)*op_p-WPR(n)*wp_p-WINJ(n)*winj_p-SINJ(n)*surf_p;
%     Inner(n)=Ct(n)/((1+r/100)^TIME_YEARS(n));
    Inner(n)=Ct(n)/((1+r/(100*365))^TIME_DAYS1(n));
end

% NPV(n)=sum(Inner);
```

```
%delete SURFACT.RSM;
%clear OPR WPR WINJ SINJ Ct Inner NPV STAT;


NetPV=sum(Inner);
CUMOIL=FOPT(LV);
clear OPR WPR WINJ SINJ Ct Inner NPV STAT;
% for t=1:TSTEP
% delete (['PRO_CONTROLS',num2str(t),'.IN'])
% delete (['INJ_CONTROLS',num2str(t),'.IN'])
% delete (['CONC_CONTROLS',num2str(t),'.IN'])
% end
end
```

REALIZATION OPTIMIZATION

This program performs the optimization for all the ensemble member of controls by using the optimal weighting factor.

realization_opt.m

```
function [Y,ALPHAstore,NPValphastore,esstore] =
realization_opt(ALPHAinit,DALPHA,Cy,Y,i,M,Ny,Nt,h,Ne,n,TSTEP,DT,grp,CP1min,CP2min,CP3min,
CP4min,RINJ1min,CONCmin,CP1max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,PCON,op_p,wp_p,winj_
p,surf_p,r,ALPHAstore,NPValphastore,ALPHAlow,ALPHAhigh,maxit,es,esstore,TRANSFORM)



       disp(['Iteration #',num2str(n)]);


[ALPHAstore,NPValphastore,esstore]=alpha_algorithm_goldensearch(ALPHAlow,ALPHAhigh,maxit,
es,Cy,Y,i,M,Ny,Nt,TSTEP,DT,grp,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1max,CP2max
,CP3max,CP4max,RINJ1max,CONCmax,PCON,op_p,wp_p,winj_p,surf_p,r,n,ALPHAstore,NPValphastore
,Ne,esstore,TRANSFORM);

        ALPHA=cell2mat(ALPHAstore(n));
        sim=length(ALPHA);

    for i=1:Ne
        disp(['Optimization for Realization #',num2str(i)]);
        if sim==1
        yk(1:Ny,i)=(1/ALPHA)*Cy*M'+Y(:,i);
        end

        if sim>1


        yk(1:Ny,i)=(1/ALPHA(sim))*Cy*M'+Y(:,i);

        end


        xk(1:Nt,i)=yk(1:Nt,i);
```

```
[xkt]=transform_x(xk,TSTEP,grp,i,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1max,CP2m
ax,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM);
        [NPVk1(i),
Poilk1(i)]=surfblackbox1(TSTEP,DT,grp,i,xkt,PCON,op_p,wp_p,winj_p,surf_p,r);


            Y(:,i)=[xk(:,i);NPVk1(i)];
    end

end
```

OPTIMIAL ALPHA ALGORITHM PROGRAM

This program determines the optimal weighting factor using the Golden Section Search alogrithm.

alpha_algorithm_goldensearch.m

```
function
[ALPHAstore,NPValphastore,esstore]=alpha_algorithm_goldensearch(xlow,xhigh,maxit,es,Cy,Y,
i,M,Ny,Nt,TSTEP,DT,grp,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1max,CP2max,CP3max,
CP4max,RINJ1max,CONCmax,PCON,op_p,wp_p,winj_p,surf_p,r,n,ALPHAstore,NPValphastore,Ne,esst
ore,TRANSFORM)


R=(5^.5-1)/2;
xl=xlow;
xu=xhigh;
d=R*(xu-xl);
iter=1;
disp(['ALPHA Determiniation Iteration #',num2str(iter)]);


x1=xl+d;
x2=xu-d;
%%
for i=1:Ne
yrunx1(1:Ny,i)=(1/x1)*Cy*M'+Y(:,i);
xrunx1(1:Nt,i)=yrunx1(1:Nt,i);
end

for i=1:Ne
[xrunx1t]=transform_x(xrunx1,TSTEP,grp,i,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1
max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM);
[NPVx1(i),
Poilx1(i)]=surfblackbox1(TSTEP,DT,grp,i,xrunx1t,PCON,op_p,wp_p,winj_p,surf_p,r); %change
end
f1=mean(NPVx1);

%%
for i=1:Ne
yrunx2(1:Ny,i)=(1/x2)*Cy*M'+Y(:,i);
xrunx2(1:Nt,i)=yrunx2(1:Nt,i);
end

for i=1:Ne
[xrunx2t]=transform_x(xrunx2,TSTEP,grp,i,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1
max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM);
[NPVx2(i),
Poilx2(i)]=surfblackbox1(TSTEP,DT,grp,i,xrunx2t,PCON,op_p,wp_p,winj_p,surf_p,r); %change
```

```
end
f2=mean(NPVx2);

%%

if f1>f2
    xopt(iter)=x1;
    fx(iter)=f1;
else
    xopt(iter)=x2;
    fx(iter)=f2;
end
ea(iter)=(1-R)*abs((xu-xl)/xopt(iter))*100
while (1)
    d=R*d;
    if f1>f2
        xl=x2;
        x2=x1;
        x1=xl+d;
        f2=f1;

        %%
        for i=1:Ne
        yrunx1(1:Ny,i)=(1/x1)*Cy*M'+Y(:,i);
        xrunx1(1:Nt,i)=yrunx1(1:Nt,i);
        end
        for i=1:Ne

[xrunx1t]=transform_x(xrunx1,TSTEP,grp,i,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1
max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM);
        [NPVx1(i),
Poilx1(i)]=surfblackbox1(TSTEP,DT,grp,i,xrunx1t,PCON,op_p,wp_p,winj_p,surf_p,r);
        end
        f1=mean(NPVx1);

    else
        xu=x1;
        x1=x2;
        x2=xu-d;
        f1=f2;
        %%
        for i=1:Ne
        yrunx2(1:Ny,i)=(1/x2)*Cy*M'+Y(:,i);
        xrunx2(1:Nt,i)=yrunx2(1:Nt,i);
        end
        for i=1:Ne

[xrunx2t]=transform_x(xrunx2,TSTEP,grp,i,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1
max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM);
        [NPVx2(i),
Poilx2(i)]=surfblackbox1(TSTEP,DT,grp,i,xrunx2t,PCON,op_p,wp_p,winj_p,surf_p,r);
        end
        f2=mean(NPVx2);

    end
    iter=iter+1;
    disp(['ALPHA Determiniation Iteration #',num2str(iter)]);
    if f1>f2
        xopt(iter)=x1;
        fx(iter)=f1;
    else
        xopt(iter)=x2;
        fx(iter)=f2;
    end
    if xopt(iter)~=0
        ea(iter)=(1-R)*abs((xu-xl)/xopt(iter))*100
    end
```

```
    if ea(iter)<=es
        break
    end
    if iter>=maxit
        break
    end

end

ALPHAstore(n)={xopt};
NPValphastore(n)={fx};
esstore(n)={ea};
end
```

SURFACTANT/RESERVOIR PROPERTIES PROGRAM

This program writes the initial reservoir properties to Eclipse and writes the surfactant properties to Eclipse.

perm_por.m

```
function
[PERM,PERM1,PERM2,PERM3,PERM4,Perm,PORO,PORO1,PORO2,PORO3,PORO4,por,Sw,So,kro,krw,Pc]=per
m_por(PERM_M,PORO_M,S_w,Pi,Swir,Sor,kroe,krwe,no,nw,Pd,PRS_INDEX,TLENGTH,Sorm_min,Sorm_ma
x,Swrm_min,Swrm_max,krom_min,krom_max,krwm_min,krwm_max)
fid5 = fopen(['PERM.IN'], 'w+t');
fprintf(fid5, 'PERMX');
fprintf(fid5, '\n');
for w=1:length(PERM_M)
fprintf(fid5, num2str(PERM_M(w,:)));
fprintf(fid5, '\n');
end
fprintf(fid5, '/');
fprintf(fid5, '\n');
fclose(fid5);

fid6 = fopen(['PORO.IN'], 'w+t');
fprintf(fid6, 'PORO');
fprintf(fid6, '\n');
for w=1:length(PORO_M)
fprintf(fid6, num2str(PORO_M(w,:)));
fprintf(fid6, '\n');
end
fprintf(fid6, '/');
fprintf(fid6, '\n');
fclose(fid6);

fid7 = fopen(['PRESSURE.IN'], 'w+t');
fprintf(fid7, 'PRESSURE');
fprintf(fid7, '\n');
fprintf(fid7, ['',num2str(50^2),'*',num2str(Pi),'/']);
fprintf(fid7, '\n');
fclose(fid7);

fid8 = fopen(['SWAT.IN'], 'w+t');
fprintf(fid8, 'SWAT');
fprintf(fid8, '\n');
fprintf(fid8, ['',num2str(50^2),'*',num2str(S_w),'/']);
fprintf(fid8, '\n');
fclose(fid8);
```

```
for l=1:TLENGTH
    if l==1
        Sw(l)=Swir;
    end
    if l>1
        Sw(l)=Sw(l-1)+0.0545;
    end
    So(l)=1-Sw(l);
    kro(l)=kroe*((1-Sw(l)-Sor)/(1-Swir-Sor))^no;
    krw(l)=krwe*((Sw(l)-Swir)/(1-Swir-Sor))^nw;
    if l==1
%      Pc(l)=Pd*((Sw(l)+.01-Swir)/(1-Swir-Sor))^(-1/PRS_INDEX);
    Pc(l)=0;
    end
    if l>1
%         Pc(l)=Pd*((Sw(l)-Swir)/(1-Swir-Sor))^(-1/PRS_INDEX);
       Pc(l)=0;
    end
end

fid9 = fopen(['SWFN.IN'], 'w+t');
fprintf(fid9, 'SWFN');
fprintf(fid9, '\n');
for l=1:TLENGTH
fprintf(fid9, ['',num2str(Sw(l)),'    ',num2str(krw(l)),'    ',num2str(Pc(l))]);
fprintf(fid9, '\n');
end
fprintf(fid9, '/');
fprintf(fid9, '\n');
% fprintf(fid9, ['',num2str(Swrm_min),'    ',num2str(krwm_min),'
',num2str(Pd*((Swrm_min+.01-Swir)/(1-Swir-Sor))^(-1/PRS_INDEX))]);
fprintf(fid9, ['',num2str(Swrm_min),'    ',num2str(krwm_min),'    ',num2str(0)]);
fprintf(fid9, '\n');
% fprintf(fid9, ['',num2str(Swrm_max),'    ',num2str(krwm_max),'
',num2str(Pd*((Swrm_max-Swir)/(1-Swir-Sor))^(-1/PRS_INDEX))]);
fprintf(fid9, ['',num2str(Swrm_max),'    ',num2str(krwm_max),'    ',num2str(0)]);
fprintf(fid9, '\n');
fprintf(fid9, '/');
fprintf(fid9, '\n');
fclose(fid9);

fid10 = fopen(['SOF2.IN'], 'w+t');
fprintf(fid10, 'SOF2');
fprintf(fid10, '\n');
for l=1:TLENGTH
fprintf(fid10, ['',num2str(So(TLENGTH+1-l)),'    ',num2str(kro(TLENGTH+1-l))]);
fprintf(fid10, '\n');
end
fprintf(fid10, '/');
fprintf(fid10, '\n');
fprintf(fid10, ['',num2str(Sorm_min),'    ',num2str(krom_min)]);
fprintf(fid10, '\n');
fprintf(fid10, ['',num2str(Sorm_max),'    ',num2str(krom_max)]);
fprintf(fid10, '\n');
fprintf(fid10, '/');
fprintf(fid10, '\n');
fclose(fid10);


PERM=PERM_M;
PORO=PORO_M;

Perm=mean(PERM); %md average permeability
por=mean(PORO); % average porosity

PERM_GRID=reshape (PERM, 50, 50)';
% imagesc (PERM_GRID,[1 1E3]);
```

```
PERM_GRID1=PERM_GRID(1:25,1:25);
PERM_GRID2=PERM_GRID(25:50,1:25);
PERM_GRID3=PERM_GRID(25:50,25:50);
PERM_GRID4=PERM_GRID(1:25,25:50);
PERM1=mean(mean(PERM_GRID1));
PERM2=mean(mean(PERM_GRID2));
PERM3=mean(mean(PERM_GRID3));
PERM4=mean(mean(PERM_GRID4));

PORO_GRID=reshape (PORO, 50, 50)';
% imagesc (PORO_GRID,[0 .5]);

PORO_GRID1=PORO_GRID(1:25,1:25);
PORO_GRID2=PORO_GRID(25:50,1:25);
PORO_GRID3=PORO_GRID(25:50,25:50);
PORO_GRID4=PORO_GRID(1:25,25:50);
PORO1=mean(mean(PORO_GRID1));
PORO2=mean(mean(PORO_GRID2));
PORO3=mean(mean(PORO_GRID3));
PORO4=mean(mean(PORO_GRID4));
end
```

GEOLOGICAL PROPERTIES FOR MONTE CARLO SAMPLING PROGRAM

This program writes the permeability and porosity field to Eclipse for Monte Carlo sampling.

perm_por2.m

```
function [Perm,por]=perm_por2(PERM_M,PORO_M)
fid5 = fopen(['PERM.IN'], 'w+t');
fprintf(fid5, 'PERMX');
fprintf(fid5, '\n');
for w=1:length(PERM_M)
fprintf(fid5, num2str(PERM_M(w,:)));
fprintf(fid5, '\n');
end
fprintf(fid5, '/');
fprintf(fid5, '\n');
fclose(fid5);

fid6 = fopen(['PORO.IN'], 'w+t');
fprintf(fid6, 'PORO');
fprintf(fid6, '\n');
for w=1:length(PORO_M)
fprintf(fid6, num2str(PORO_M(w,:)));
fprintf(fid6, '\n');
end
fprintf(fid6, '/');
fprintf(fid6, '\n');
fclose(fid6);




PERM=PERM_M;
PORO=PORO_M;
```

```
Perm=mean(PERM); %md average permeability
por=mean(PORO); % average porosity


end
```

TRANSFORM PROGRAM 1

This program transforms one ensemble member of control vector x so that Eclipse may use the control.

transform_x.m

```
function
[xconv]=transform_x(xtrans,TSTEP,grp,i,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP1ma
x,CP2max,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM)

if TRANSFORM==1
CP1mu=(CP1min+CP1max)/2;
CP1stdv=(CP1max-CP1min)/2;

CP2mu=(CP2min+CP2max)/2;
CP2stdv=(CP2max-CP2min)/2;

CP3mu=(CP3min+CP3max)/2;
CP3stdv=(CP3max-CP3min)/2;

CP4mu=(CP4min+CP4max)/2;
CP4stdv=(CP4max-CP4min)/2;

RINJmu=(RINJ1max+RINJ1min)/2;
RINJstdv=(RINJ1max-RINJ1min)/2;

CONCmu=(CONCmax+CONCmin)/2;
CONCstdv=(CONCmax-CONCmin)/2;

for t=1:TSTEP

    PROB(grp(t)-5,i)=.5*(1+erf(xtrans(grp(t)-5,i)/sqrt(2)));
    if PROB(grp(t)-5,i)==0
        xconv(grp(t)-5,i)=CP1min;
    elseif PROB(grp(t)-5,i)==1
        xconv(grp(t)-5,i)=CP1max;
    else xconv(grp(t)-5,i)=PROB(grp(t)-5,i)*(CP1max-CP1min)+CP1min;
    end


    PROB(grp(t)-4,i)=.5*(1+erf(xtrans(grp(t)-4,i)/sqrt(2)));
    if PROB(grp(t)-4,i)==0
        xconv(grp(t)-4,i)=CP2min;
    elseif PROB(grp(t)-4,i)==1
        xconv(grp(t)-4,i)=CP2max;
    else xconv(grp(t)-4,i)=PROB(grp(t)-4,i)*(CP2max-CP2min)+CP2min;
    end


    PROB(grp(t)-3,i)=.5*(1+erf(xtrans(grp(t)-3,i)/sqrt(2)));
```

```
    if PROB(grp(t)-3,i)==0
        xconv(grp(t)-3,i)=CP3min;
    elseif PROB(grp(t)-3,i)==1
        xconv(grp(t)-3,i)=CP3max;
    else xconv(grp(t)-3,i)=PROB(grp(t)-3,i)*(CP3max-CP3min)+CP3min;
    end

    PROB(grp(t)-2,i)=.5*(1+erf(xtrans(grp(t)-2,i)/sqrt(2)));
    if PROB(grp(t)-2,i)==0
        xconv(grp(t)-2,i)=CP4min;
    elseif PROB(grp(t)-2,i)==1
        xconv(grp(t)-2,i)=CP4max;
    else xconv(grp(t)-2,i)=PROB(grp(t)-2,i)*(CP4max-CP4min)+CP4min;
    end


    PROB(grp(t)-1,i)=.5*(1+erf(xtrans(grp(t)-1,i)/sqrt(2)));
    if PROB(grp(t)-1,i)==0
        xconv(grp(t)-1,i)=RINJ1min;
    elseif PROB(grp(t)-1,i)==1
        xconv(grp(t)-1,i)=RINJ1max;
    else xconv(grp(t)-1,i)=PROB(grp(t)-1,i)*(RINJ1max-RINJ1min)+RINJ1min;
    end


    PROB(grp(t),i)=.5*(1+erf(xtrans(grp(t),i)/sqrt(2)));
    if PROB(grp(t),i)==0
        xconv(grp(t),i)=CONCmin;
    elseif PROB(grp(t),i)==1
        xconv(grp(t),i)=CONCmax;
    else xconv(grp(t),i)=PROB(grp(t),i)*(CONCmax-CONCmin)+CONCmin;
    end



end
end

if TRANSFORM==0
for t=1:TSTEP
    xconv(grp(t)-5,i)=exp(xtrans(grp(t)-5,i));

    xconv(grp(t)-4,i)=exp(xtrans(grp(t)-4,i));

    xconv(grp(t)-3,i)=exp(xtrans(grp(t)-3,i));

    xconv(grp(t)-2,i)=exp(xtrans(grp(t)-2,i));

    xconv(grp(t)-1,i)=exp(xtrans(grp(t)-1,i));

    xconv(grp(t),i)=exp(xtrans(grp(t),i));

end
end

end
```

TRANSFORM PROGRAM 2

This program transforms all ensemble member of control vectors x so that Eclipse may use the control.

transform_xNe.m

```
function
[xconv]=transform_xNe(xtrans,TSTEP,grp,Ne,CP1min,CP2min,CP3min,CP4min,RINJ1min,CONCmin,CP
1max,CP2max,CP3max,CP4max,RINJ1max,CONCmax,TRANSFORM)

for i=1:Ne
if TRANSFORM==1
CP1mu=(CP1min+CP1max)/2;
CP1stdv=(CP1max-CP1min)/2;

CP2mu=(CP2min+CP2max)/2;
CP2stdv=(CP2max-CP2min)/2;

CP3mu=(CP3min+CP3max)/2;
CP3stdv=(CP3max-CP3min)/2;

CP4mu=(CP4min+CP4max)/2;
CP4stdv=(CP4max-CP4min)/2;

RINJmu=(RINJ1max+RINJ1min)/2;
RINJstdv=(RINJ1max-RINJ1min)/2;

CONCmu=(CONCmax+CONCmin)/2;
CONCstdv=(CONCmax-CONCmin)/2;

for t=1:TSTEP

    PROB(grp(t)-5,i)=.5*(1+erf(xtrans(grp(t)-5,i)/sqrt(2)));
    if PROB(grp(t)-5,i)==0
        xconv(grp(t)-5,i)=CP1min;
    elseif PROB(grp(t)-5,i)==1
        xconv(grp(t)-5,i)=CP1max;
    else xconv(grp(t)-5,i)=PROB(grp(t)-5,i)*(CP1max-CP1min)+CP1min;
    end


    PROB(grp(t)-4,i)=.5*(1+erf(xtrans(grp(t)-4,i)/sqrt(2)));
    if PROB(grp(t)-4,i)==0
        xconv(grp(t)-4,i)=CP2min;
    elseif PROB(grp(t)-4,i)==1
        xconv(grp(t)-4,i)=CP2max;
    else xconv(grp(t)-4,i)=PROB(grp(t)-4,i)*(CP2max-CP2min)+CP2min;
    end


    PROB(grp(t)-3,i)=.5*(1+erf(xtrans(grp(t)-3,i)/sqrt(2)));
    if PROB(grp(t)-3,i)==0
        xconv(grp(t)-3,i)=CP3min;
    elseif PROB(grp(t)-3,i)==1
        xconv(grp(t)-3,i)=CP3max;
    else xconv(grp(t)-3,i)=PROB(grp(t)-3,i)*(CP3max-CP3min)+CP3min;
    end


    PROB(grp(t)-2,i)=.5*(1+erf(xtrans(grp(t)-2,i)/sqrt(2)));
    if PROB(grp(t)-2,i)==0
        xconv(grp(t)-2,i)=CP4min;
    elseif PROB(grp(t)-2,i)==1
        xconv(grp(t)-2,i)=CP4max;
    else xconv(grp(t)-2,i)=PROB(grp(t)-2,i)*(CP4max-CP4min)+CP4min;
    end


    PROB(grp(t)-1,i)=.5*(1+erf(xtrans(grp(t)-1,i)/sqrt(2)));
    if PROB(grp(t)-1,i)==0
        xconv(grp(t)-1,i)=RINJ1min;
```

```
    elseif PROB(grp(t)-1,i)==1
        xconv(grp(t)-1,i)=RINJ1max;
    else xconv(grp(t)-1,i)=PROB(grp(t)-1,i)*(RINJ1max-RINJ1min)+RINJ1min;
    end


    PROB(grp(t),i)=.5*(1+erf(xtrans(grp(t),i)/sqrt(2)));
    if PROB(grp(t),i)==0
        xconv(grp(t),i)=CONCmin;
    elseif PROB(grp(t),i)==1
        xconv(grp(t),i)=CONCmax;
    else xconv(grp(t),i)=PROB(grp(t),i)*(CONCmax-CONCmin)+CONCmin;
    end



end
end

if TRANSFORM==0
for t=1:TSTEP
    xconv(grp(t)-5,i)=exp(xtrans(grp(t)-5,i));

    xconv(grp(t)-4,i)=exp(xtrans(grp(t)-4,i));

    xconv(grp(t)-3,i)=exp(xtrans(grp(t)-3,i));

    xconv(grp(t)-2,i)=exp(xtrans(grp(t)-2,i));

    xconv(grp(t)-1,i)=exp(xtrans(grp(t)-1,i));

    xconv(grp(t),i)=exp(xtrans(grp(t),i));

end
end
end
end
```

GRAPHICS PROGRAM

This program is run separately because it requires all lot of memory. It is best to load the final file from the Main program and run this graphic program. This program generates useful graphs for analysis purposes. This program also saves the plots in .tif format.

ProjectGraphics.m

```
close all;
%Original Control and Optimized Control From Matrix
for i=1:Graph
for t=1:TSTEP
    C1plot(t)=xoptimalt(grp(t)-5,i); %change
    C2plot(t)=xoptimalt(grp(t)-4,i); %change
    C3plot(t)=xoptimalt(grp(t)-3,i); %change
    C4plot(t)=xoptimalt(grp(t)-2,i); %change
    C5plot(t)=xoptimalt(grp(t)-1,i); %change
    C6plot(t)=xoptimalt(grp(t),i); %change
```

```
    C1orgplot(t)=xorgt(grp(t)-5,i); %change
    C2orgplot(t)=xorgt(grp(t)-4,i); %change
    C3orgplot(t)=xorgt(grp(t)-3,i); %change
    C4orgplot(t)=xorgt(grp(t)-2,i); %change
    C5orgplot(t)=xorgt(grp(t)-1,i); %change
    C6orgplot(t)=xorgt(grp(t),i); %change
end
end

%Original Control and Optimized Control From Matrix for all realizations
for i=1:Ne
for t=1:TSTEP
    C1plotall(t,i)=xoptimalt(grp(t)-5,i); %change
    C2plotall(t,i)=xoptimalt(grp(t)-4,i); %change
    C3plotall(t,i)=xoptimalt(grp(t)-3,i); %change
    C4plotall(t,i)=xoptimalt(grp(t)-2,i); %change
    C5plotall(t,i)=xoptimalt(grp(t)-1,i); %change
    C6plotall(t,i)=xoptimalt(grp(t),i); %change
    C1orgplotall(t,i)=xorgt(grp(t)-5,i); %change
    C2orgplotall(t,i)=xorgt(grp(t)-4,i); %change
    C3orgplotall(t,i)=xorgt(grp(t)-3,i); %change
    C4orgplotall(t,i)=xorgt(grp(t)-2,i); %change
    C5orgplotall(t,i)=xorgt(grp(t)-1,i); %change
    C6orgplotall(t,i)=xorgt(grp(t),i); %change
end
end

%Iterations vs Average NPV
figure;
plot([0,it(1:n-1)],[mean(NPVinit),NPVmeanit(1:n-1)]/10^6,'-ko','MarkerFaceColor','k');
xlabel('iteration');
ylabel('Average Net Present Value, MM$');
title('Iterations versus Average Net Present Value');
saveas(gcf,'Iterations versus Average Net Present Value.tif')
% print -f1 -djpeg90 PERM_ITER.tif

%Iterations vs select Realizations of NPV
for i=1:Graph
figure;
plot([0,it(1:n-1)],[NPVinit(i);NPVreal(1:n-1,i)]/10^6,'-ko','MarkerFaceColor','k');
xlabel('iteration');
ylabel('Net Present Value, MM$');
title(['Iterations versus NPV for Realization #', num2str(i)]);
saveas(gcf,['Iterations versus NPV for Realization #', num2str(i),'.tif'])
end

%Iterations vs all Realizations of NPV
figure;
plot([0,it(1:n-1)],[NPVinit;NPVreal(1:n-1,:)]/10^6);
xlabel('iteration');
ylabel('Net Present Value, MM$');
title(['Iterations versus NPV for All Realization']);
saveas(gcf,'Iterations versus NPV for All Realization.tif')

%Iterations vs STDV
figure;
plot([0,it(1:n-1)],[std(NPVinit);std(NPVreal(1:n-1,:),1,2)],'-ko','MarkerFaceColor','k');
xlabel('iteration');
ylabel('Net Present Value, $');
title(['Iterations versus Standard Devation of all NPV Realization']);
saveas(gcf,'Iterations versus Standard Devation of all NPV Realization.tif')

%plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FOPTinitf(i)),'Color',[1 0 0]);
figure;
bar(Poilinitf/10^3);
xlabel('Realization');
```

```
ylabel('Cumulative Oil, MSTB');
title('Inital Cumulative Oil');
saveas(gcf,'Inital Cumulative Oil.tif')

figure;
bar(Poiloptf/10^3);
xlabel('Realization');
ylabel('Cumulative Oil, MSTB');
title('Optimized Cumulative Oil');
saveas(gcf,'Optimized Cumulative Oil.tif')

figure;
bar(Poilincrease);
xlabel('Realization');
ylabel('Increase Cumulative Oil, %');
title('Increase Cumulative Oil');
saveas(gcf,'Increase Cumulative Oil.tif')

%%
figure;
bar(NPVinit/10^6);
xlabel('Realization');
ylabel('NPV, MM$');
title('Original NPV distribution');
saveas(gcf,'Original NPV distribution.tif')

figure;
bar(NPVreal(n-1,:)/10^6);
xlabel('Realization');
ylabel('NPV, MM$');
title('Optimized NPV distribution');
saveas(gcf,'Optimized NPV distribution.tif')

figure;
bar(NPVincrease);
xlabel('Realization');
ylabel('Increase in NPV, %');
title('Increase in NPV');
saveas(gcf,'Increase in NPV.tif')
%%

figure;
bar(RFinit);
xlabel('Realization');
ylabel('Inital Recovery Factor, %');
title('Inital Recovery Factor');
saveas(gcf,'Inital Recovery Factor.tif')

figure;
bar(RFopt);
xlabel('Realization');
ylabel('Optimized Recovery Factor, %');
title('Optimized Recovery Factor');
saveas(gcf,'Optimized Recovery Factor.tif')

figure;
bar(RFincrease);
xlabel('Realization');
ylabel('Recovery Factor Increase, %');
title('Recovery Factor Increase');
saveas(gcf,'Recovery Factor Increase.tif')

%Original Control Plots
for i=1:Graph

figure;
if PCON==0
```

```
stairs([0,period],[C1orgplot,C1orgplot(TSTEP)],'-r');
hold on
stairs([0,period],[C2orgplot,C2orgplot(TSTEP)],'-g');
hold on
stairs([0,period],[C3orgplot,C3orgplot(TSTEP)],'-b');
hold on
stairs([0,period],[C4orgplot,C4orgplot(TSTEP)],'-k');
xlabel('Time, Months');
ylabel('PSIA');
title(['Original BHP Profile for Realization #',num2str(i)]);
legend('P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside')
saveas(gcf,['Original BHP Profile for Realization #', num2str(i),'.tif'])
end


if PCON==1
stairs([0,period],[C1orgplot,C1orgplot(TSTEP)],'-r');
hold on
stairs([0,period],[C2orgplot,C2orgplot(TSTEP)],'-g');
hold on
stairs([0,period],[C3orgplot,C3orgplot(TSTEP)],'-b');
hold on
stairs([0,period],[C4orgplot,C4orgplot(TSTEP)],'-k');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Oil Flowrate Profile for Realization #',num2str(i)]);
legend('P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside')
saveas(gcf,['Original Oil Flowrate Profile for Realization #', num2str(i),'.tif'])
end


if PCON==2
stairs([0,period],[C1orgplot,C1orgplot(TSTEP)],'-r');
hold on
stairs([0,period],[C2orgplot,C2orgplot(TSTEP)],'-g');
hold on
stairs([0,period],[C3orgplot,C3orgplot(TSTEP)],'-b');
hold on
stairs([0,period],[C4orgplot,C4orgplot(TSTEP)],'-k');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Total Flowrate Profile for Realization #',num2str(i)]);
legend('P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside')
saveas(gcf,['Original Total Flowrate Profile for Realization #', num2str(i),'.tif'])
end

figure;
stairs([0,period],[C5orgplot,C5orgplot(TSTEP)],'-k');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Injection Rate Profile for Realization #',num2str(i)]);
saveas(gcf,['Original Injection Rate Profile for Realization #', num2str(i),'.tif'])

figure;
stairs([0,period],[C6orgplot,C6orgplot(TSTEP)],'-k');
xlabel('Time, Months');
ylabel('LB/STB');
title(['Original Concentration of Surfactant Injected Profile for Realization
#',num2str(i)]);
saveas(gcf,['Original Concentration of Surfactant Injected Profile for Realization #',
num2str(i),'.tif'])
end

%Control Comparison Plots between Original and Optimized
for i=1:Graph

figure;
```

```
stairs([0,period],[C1plot,C1plot(TSTEP)],'-b');
hold on
stairs([0,period],[C1orgplot,C1orgplot(TSTEP)],'-r');
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized BHP Profile for Well P1 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized BHP Profile for Well P1 Realization #', num2str(i),'.tif'])

end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Total Flowrate Profile for Well P1 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized Total Flowrate Profile for Well P1 Realization #',
num2str(i),'.tif'])
end



figure;
stairs([0,period],[C2plot,C2plot(TSTEP)],'-b');
hold on
stairs([0,period],[C2orgplot,C2orgplot(TSTEP)],'-r');
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized BHP Profile for Well P2 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized BHP Profile for Well P2 Realization #', num2str(i),'.tif'])
end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Total Flowrate Profile for Well P2 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized Total Flowrate Profile for Well P2 Realization #',
num2str(i),'.tif'])
end


figure;
stairs([0,period],[C3plot,C3plot(TSTEP)],'-b');
hold on
stairs([0,period],[C3orgplot,C3orgplot(TSTEP)],'-r');
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized BHP Profile for Well P3 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized BHP Profile for Well P3 Realization #', num2str(i),'.tif'])
end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Total Flowrate Profile for Well P3 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized Total Flowrate Profile for Well P3 Realization #',
num2str(i),'.tif'])
end


figure;
stairs([0,period],[C4plot,C4plot(TSTEP)],'-b');
hold on
```

```
stairs([0,period],[C4orgplot,C4orgplot(TSTEP)],'-r');
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized BHP Profile for Well P4 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized BHP Profile for Well P4 Realization #', num2str(i),'.tif'])
end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Total Flowrate Profile for Well P4 Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized Total Flowrate Profile for Well P4 Realization #',
num2str(i),'.tif'])
end


figure;
stairs([0,period],[C5plot,C5plot(TSTEP)],'-b');
hold on
stairs([0,period],[C5orgplot,C5orgplot(TSTEP)],'-r');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Injection Rate Profile for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized Injection Rate Profile for Realization #', num2str(i),'.tif'])

figure;
stairs([0,period],[C6plot,C6plot(TSTEP)],'-b');
hold on
stairs([0,period],[C6orgplot,C6orgplot(TSTEP)],'-r');
xlabel('Time, Months');
ylabel('LB/STB');
title(['Optimized Concentration of Surfactant Injected Profile for Realization
#',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Optimized Concentration of Surfactant Injected Profile for Realization #',
num2str(i),'.tif'])
end

%Property Comparison Plots between Original and Optimized (Eclipse RSM File)
for i=1:Graph
figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(FWCToptf(i)),'-
ok','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT1optf(i)),'-
sk','MarkerFaceColor','r');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT2optf(i)),'-
dk','MarkerFaceColor','g');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT3optf(i)),'-
pk','MarkerFaceColor','k');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT4optf(i)),'-
^k','MarkerFaceColor','m');
xlabel('Time, Months');
ylabel('Watercut,%');
legend('Field','P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside');
title(['Optimized Watercut Profile for Realization #',num2str(i)]);
saveas(gcf,['Optimized Watercut Profile for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(FWCTinitf(i)),'-
ok','MarkerFaceColor','b');
```

```
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT1initf(i)),'-
sk','MarkerFaceColor','r');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT2initf(i)),'-
dk','MarkerFaceColor','g');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT3initf(i)),'-
pk','MarkerFaceColor','k');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT4initf(i)),'-
^k','MarkerFaceColor','m');
xlabel('Time, Months');
ylabel('Watercut,%');
legend('Field','P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside');
title(['Original Watercut Profile for Realization #',num2str(i)]);
saveas(gcf,['Original Watercut Profile for Realization #', num2str(i),'.tif'])


figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FOPToptf(i))/10^3,'-
ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FOPTinitf(i))/10^3,'-
ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('MSTB');
title(['Cumulative Oil Production for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Cumulative Oil Production for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FWPToptf(i))/10^3,'-
ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FWPTinitf(i))/10^3,'-
ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('MSTB');
title(['Cumulative Water Production for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Cumulative Water Production for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(FOSAToptf(i)),'-
ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(FOSATinitf(i)),'-
ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('So,%');
title(['Field Oil Saturation for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Field Oil Saturation for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(FPRoptf(i)),'-ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(FPRinitf(i)),'-ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('PSIA');
title(['Field Pressure Profile for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Field Pressure Profile for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP1optf(i)),'-ok','MarkerFaceColor','r');
```

```
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP2optf(i)),'-sk','MarkerFaceColor','g');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP3optf(i)),'-dk','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP4optf(i)),'-^k','MarkerFaceColor','k');
xlabel('Time, Months');
ylabel('PSIA');
legend('P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside');
title(['Optimized Bottom Hole Pressure Profile for Realization #',num2str(i)]);
saveas(gcf,['Optimized Bottom Hole Pressure Profile for Realization #',
num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP1initf(i)),'-
ok','MarkerFaceColor','r');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP2initf(i)),'-
sk','MarkerFaceColor','g');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP3initf(i)),'-
dk','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP4initf(i)),'-
^k','MarkerFaceColor','k');
xlabel('Time, Months');
ylabel('PSIA');
legend('P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside');
title(['Original Bottom Hole Pressure Profile for Realization #',num2str(i)]);
saveas(gcf,['Original Bottom Hole Pressure Profile for Realization #',
num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP1optf(i)),'-ok','MarkerFaceColor','r');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP2optf(i)),'-sk','MarkerFaceColor','g');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP3optf(i)),'-dk','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP4optf(i)),'-^k','MarkerFaceColor','k');
xlabel('Time, Months');
ylabel('STD/DAY');
legend('P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside');
title(['Optimized Liquid Production Profile for Realization #',num2str(i)]);
saveas(gcf,['Optimized Liquid Production Profile for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP1initf(i)),'-ok','MarkerFaceColor','r');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP2initf(i)),'-sk','MarkerFaceColor','g');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP3initf(i)),'-dk','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP4initf(i)),'-^k','MarkerFaceColor','k');
xlabel('Time, Months');
ylabel('STD/DAY');
legend('P1','P2','P3','P4','Orientation','horizontal','Location','SouthOutside');
title(['Original Liquid Production Profile for Realization #',num2str(i)]);
saveas(gcf,['Original Liquid Production Profile for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(RINJoptf(i)),'-ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(RINJinitf(i)),'-ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Injection Rate Profile for Realization #',num2str(i)]);
```

```
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Injection Rate Profile for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(CONCoptf(i)),'-ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(CONCinitf(i)),'-ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('LB/STB');
title(['Injected Surfactant Concentration Profile for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Injected Surfactant Concentration Profile for Realization #',
num2str(i),'.tif'])
end

%% All Original and Optimized Input Controls
figure;
stairs([0,period],[C1plotall;C1plotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P1 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P1 BHP Profile for All ',num2str(Ne),' Realizations.tif'])
end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P1 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations']);
saveas(gcf,['Optimized Well P1 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])
end

figure;
stairs([0,period],[C1orgplotall;C1orgplotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P1 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P1 BHP Profile for All ',num2str(Ne),' Realizations.tif'])
end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Well P1 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P1 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])
end


figure;
stairs([0,period],[C2plotall;C2plotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P2 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P2 BHP Profile for All ',num2str(Ne),' Realizations.tif'])
end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P2 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations']);
saveas(gcf,['Optimized Well P2 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])
end
```

```
figure;
stairs([0,period],[C2orgplotall;C2orgplotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P2 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P2 BHP Profile for All ',num2str(Ne),' Realizations.tif'])
end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Well P2 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P2 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])
end


figure;
stairs([0,period],[C3plotall;C3plotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P3 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P3 BHP Profile for All ',num2str(Ne),' Realizations.tif'])

end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P3 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations']);
saveas(gcf,['Optimized Well P3 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])

end

figure;
stairs([0,period],[C3orgplotall;C3orgplotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P3 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P3 BHP Profile for All ',num2str(Ne),' Realizations.tif'])

end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Orignal Well P3 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Orignal Well P3 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])

end

figure;
stairs([0,period],[C4plotall;C4plotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P4 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P4 BHP Profile for All ',num2str(Ne),' Realizations.tif'])

end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P4 Liquid Flowrate Profile for All ',num2str(Ne),'
```

```
Realizations']);
saveas(gcf,['Optimized Well P4 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])

end

figure;
stairs([0,period],[C4orgplotall;C4orgplotall(TSTEP,:)]);
if PCON==0
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P4 BHP Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P4 BHP Profile for All ',num2str(Ne),' Realizations.tif'])

end
if PCON==2
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Well P4 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P4 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations.tif'])
end


figure;
stairs([0,period],[C5plotall;C5plotall(TSTEP,:)]);
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Injection Rate Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Injection Rate Profile for All ',num2str(Ne),' Realizations.tif'])

figure;
stairs([0,period],[C5orgplotall;C5orgplotall(TSTEP,:)]);
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Injection Rate Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Injection Rate Profile for All ',num2str(Ne),' Realizations.tif'])

figure;
stairs([0,period],[C6plotall;C6plotall(TSTEP,:)]);
xlabel('Time, Months');
ylabel('LB/STB');
title(['Optimized Surfactant Concentration Profile for All ',num2str(Ne),'
Realizations']);
saveas(gcf,['Optimized Surfactant Concentration Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
stairs([0,period],[C6orgplotall;C6orgplotall(TSTEP,:)]);
xlabel('Time, Months');
ylabel('LB/STB');
title(['Original Surfactant Concentration Profile for All ',num2str(Ne),'
Realizations']);
saveas(gcf,['Original Surfactant Concentration Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(FWCToptf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Optimized Field Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Field Watercut Profile for All ',num2str(Ne),' Realizations.tif'])
```

```
figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(FWCTinitf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Original Field Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Field Watercut Profile for All ',num2str(Ne),' Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT1optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Optimized Well P1 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P1 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT1initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Original Well P1 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P1 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT2optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Optimized Well P2 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P2 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT2initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Original Well P2 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P2 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
```

```
    plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT3optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Optimized Well P3 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P3 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT3initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Original Well P3 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P3 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),100*cell2mat(WCT4optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Optimized Well P4 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Well P4 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),100*cell2mat(WCT4initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Watercut,%');
title(['Original Well P4 Watercut Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Well P4 Watercut Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP1optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P1 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Optimized Well P1 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP1initf(i)));
    if i<Ne
```

```
    hold all;
    end
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P1 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Original Well P1 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP2optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P2 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Optimized Well P2 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP2initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P2 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Original Well P2 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP3optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P3 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Optimized Well P3 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP3initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P3 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Original Well P3 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(WBHP4optf(i)));
    if i<Ne
    hold all;
    end
```

```
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Optimized Well P4 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Optimized Well P4 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(WBHP4initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('PSIA');
title(['Original Well P4 BHP Profile for All ',num2str(Ne),' Realizations (Eclipse)']);
saveas(gcf,['Original Well P4 BHP Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP1optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P1 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Optimized Well P1 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP1initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Well P1 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Original Well P1 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP2optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P2 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Optimized Well P2 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP2initf(i)));
    if i<Ne
    hold all;
```

```
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Well P2 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Original Well P2 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP3optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P3 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Optimized Well P3 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP3initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Well P3 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Original Well P3 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(LP4optf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Well P4 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Optimized Well P4 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(LP4initf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Well P4 Liquid Flowrate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Original Well P4 Liquid Flowrate Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
```

```
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(RINJoptf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Optimized Injection Rate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Optimized Injection Rate Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(RINJinitf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('STB/DAY');
title(['Original Injection Rate Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Original Injection Rate Profile for All ',num2str(Ne),' Realizations
(Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(CONCoptf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('LB/STB');
title(['Optimized Surfactant Concentration Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Optimized Surfactant Concentration Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(CONCinitf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('LB/STB');
title(['Original Surfactant Concentration Profile for All ',num2str(Ne),' Realizations
(Eclipse)']);
saveas(gcf,['Original Surfactant Concentration Profile for All ',num2str(Ne),'
Realizations (Eclipse).tif'])

close all;
figure;
for k=1:n-1
    semilogx(cell2mat(ALPHAstore(k)),cell2mat(NPValphastore(k))/10^6,'-o');
    if k<n-1
    hold all;
    end
    tttt(k)=cellstr(['Opt. iteration ',num2str(k)]);
end
xlabel('ALPHA');
ylabel('NPV, MM$');
title('NPV versus ALPHA');
legend(tttt,'Orientation','vertical','Location','EastOutside');
```

```
saveas(gcf,'NPV versus ALPHA.tif')

figure;
for k=1:n-1
    plot(1:length(cell2mat(esstore(k))),cell2mat(esstore(k)),'-o');

    if k<n-1
    hold all;
    end
    ssss(k)=cellstr(['Opt. iteration ',num2str(k)]);
end
xlabel('ITERATIONS');
ylabel('ERROR, %');
title('Error in Alpha versus Iterations');
legend(ssss,'Orientation','vertical','Location','EastOutside');
saveas(gcf,'Error in Alpha versus Iterations.tif')

figure;
cdfplot(NPVprob/10^6);
xlabel('NPV, MM$');
ylabel('Probability');
title(['Probabilistic NPV for Realization #', num2str(1)]);
saveas(gcf,['Probabilistic NPV for Realization #', num2str(1),'.tif'])

figure;
cdfplot(Poilprob/10^3);
xlabel('Cumulative Production, MSTB');
ylabel('Probability');
title(['Probabilistic Cumulative Oil Production for Realization #',num2str(1)]);
saveas(gcf,['Probabilistic Cumulative Oil Production for Realization #',
num2str(1),'.tif'])

figure;
bar(op_preal);
xlabel('Realization');
ylabel('$/STB');
title('Oil Price Realizations');
saveas(gcf,'Oil Price Realizations.tif')

figure;
bar(winj_preal);
xlabel('Realization');
ylabel('$/STB');
title('Water Injection Cost Realizations');
saveas(gcf,'Water Injection Cost Realizations.tif')

figure;
bar(wp_preal);
xlabel('Realization');
ylabel('$/STB');
title('Water Treatment Cost Realizations');
saveas(gcf,'Water Treatment Cost Realizations.tif')

figure;
bar(surf_preal);
xlabel('Realization');
ylabel('$/lb');
title('Surfactant Cost Realizations');
saveas(gcf,'Surfactant Cost Realizations.tif')

for i=1:NR
figure;
imagesc (reshape(PERMALL(:,i), 50, 50)',[1 5E3]);
title(['Permeability Field Realization #',num2str(i),' (mD)']);
colorbar;
saveas(gcf,['Permeability Field Realization #',num2str(i),' (mD).tif'])
end
```

```
for i=1:NR
figure;
imagesc (reshape(POROALL(:,i), 50, 50)',[.1 .25]);
title(['Porosity Field Realization #',num2str(i)]);
colorbar;
saveas(gcf,['Porosity Field Realization #',num2str(i),'.tif'])
end

figure;
imagesc (reshape(PERM_MEAN, 50, 50)',[1 5E3]);
title('Mean Permeability Field (mD)');
colorbar;
saveas(gcf,'Mean Permeability Field.tif')


figure;
imagesc (reshape(PORO_MEAN, 50, 50)',[.1 .25]);
title('Mean Porosity Field');
colorbar;
saveas(gcf,'Mean Porosity Field.tif')

figure;
imagesc (reshape(OIP_BLOCK/(10^3), 50, 50)',[4E3/10^3 1.5E4/10^3]);
title('OIP Field Distribution, MSTB');
colorbar;
saveas(gcf,'OIP Field Distribution.tif')

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(SURFPRODToptf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('LB');
title(['Optimized Produced Surfactant Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Optimized Produced Surfactant Profile for All ',num2str(Ne),'
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(SURFPRODTinitf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('LB');
title(['Original Produced Surfactant Profile for All ',num2str(Ne),' Realizations']);
saveas(gcf,['Original Produced Surfactant Profile for All ',num2str(Ne),'
Realizations.tif'])


figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(POREINJinitf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Pore Volume');
title(['Original Injected Pore Volume of Water Profile for All ',num2str(Ne),'
Realizations']);
saveas(gcf,['Original Injected Pore Volume of Water Profile for All ',num2str(Ne),'
```

```
Realizations.tif'])

figure;
for i=1:Ne
    plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(POREINJoptf(i)));
    if i<Ne
    hold all;
    end
end
xlabel('Time, Months');
ylabel('Pore Volume');
title(['Optimized Injected Pore Volume of Water Profile for All ',num2str(Ne),'
Realizations']);
saveas(gcf,['Optimized Injected Pore Volume of Water Profile for All ',num2str(Ne),'
Realizations.tif'])

for i=1:Graph
figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(SURFPRODToptf(i)),'-
ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(SURFPRODTinitf(i)),'-
ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('LB');
title(['Produced Surfactant Profile for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Produced Surfactant Profile for Realization #', num2str(i),'.tif'])

figure;
plot(12*cell2mat(TIME_YEARSoptf(i)),cell2mat(POREINJoptf(i)),'-
ko','MarkerFaceColor','b');
hold on
plot(12*cell2mat(TIME_YEARSinitf(i)),cell2mat(POREINJinitf(i)),'-
ks','MarkerFaceColor','r');
xlabel('Time, Months');
ylabel('Pore Volume');
title(['Injected Pore Volume of Water Profile for Realization #',num2str(i)]);
legend('Optimized','Original','Orientation','horizontal','Location','SouthOutside');
saveas(gcf,['Injected Pore Volume of Water Profile for Realization #',
num2str(i),'.tif'])
end
```

COLUMN DATA READING PROGRAM

The program reads data in a column format. Program was written by Gerald Recktenwald (Recktenwald, 1995).

readColdata.mat

```
function  [labels,x,y] = readColData(fname,ncols,nhead,nlrows)
%  readColData  reads data from a file containing data in columns
%               that have text titles, and possibly other header text
%
%  Synopsis:
%     [labels,x,y] = readColData(fname)
%     [labels,x,y] = readColData(fname,ncols)
%     [labels,x,y] = readColData(fname,ncols,nhead)
```

```
%     [labels,x,y] = readColData(fname,ncols,nhead,nlrows)
%
%  Input:
%     fname  = name of the file containing the data (required)
%     ncols  = number of columns in the data file.  Default = 2.  A value
%              of ncols is required only if nlrows is also specified.
%     nhead  = number of lines of header information at the very top of
%              the file.  Header text is read and discarded.  Default = 0.
%              A value of nhead is required only if nlrows is also specified.
%     nlrows = number of rows of labels.  Default = 1
%
%  Output:
%     labels  =  matrix of labels.  Each row of lables is a different
%                label from the columns of data.  The number of columns
%                in the labels matrix equals the length of the longest
%                column heading in the data file.  More than one row of
%                labels is allowed.  In this case the second row of column
%                headings begins in row ncol+1 of labels.  The third row
%                column headings begins in row 2*ncol+1 of labels, etc.
%
%          NOTE:  Individual column headings must not contain blanks
%
%     x = column vector of x values
%     y = matrix of y values.  y has length(x) rows and ncols columns
%
%  Author:
%     Gerald Recktenwald, gerry@me.pdx.edu
%     Portland State University, Mechanical Engineering Department
%     24 August 1995

%  process optional arguments
if nargin < 4
   nlrows = 1;      % default
   if nargin < 3
      nhead = 0;      % default
      if nargin < 2
         ncols = 2;    % default
      end
   end
end

%  open file for input, include error handling
fin = fopen(fname,'r');
if fin < 0
   error(['Could not open ',fname,' for input']);
end

%  Preliminary reading of titles to determine number of columns
%  needed in the labels matrix.  This allows for an arbitrary number
%  of column titles with unequal (string) lengths.  We cannot simply
%  append to the labels matrix as new labels are read because the first
%  label might not be the longest.  The number of columns in the labels
%  matrix (= maxlen) needs to be set properly from the start.

%  Read and discard header text on line at a time
for i=1:nhead,  buffer = fgetl(fin);  end

maxlen = 0;
for i=1:nlrows
   buffer = fgetl(fin);          %  get next line as a string
   for j=1:ncols
      [next,buffer] = strtok(buffer);       %  parse next column label
      maxlen = max(maxlen,length(next));   %  find the longest so far
   end

end
```

```
%  Set the number of columns in the labels matrix equal to the length
%  of the longest column title.  A complete preallocation (including
%  rows) of the label matrix is not possible since there is no string
%  equivalent of the ones() or zeros() command.  The blank() command
%  only creates a string row vector not a matrix.
labels = blanks(maxlen);

frewind(fin);    %  rewind in preparation for actual reading of labels and data

%  Read and discard header text on line at a time
for i=1:nhead,  buffer = fgetl(fin);  end

%  Read titles for keeps this time
for i=1:nlrows

   buffer = fgetl(fin);          %  get next line as a string
   for j=1:ncols
      [next,buffer] = strtok(buffer);     %  parse next column label
      n = j + (i-1)*ncols;                %  pointer into the label array for next label
      labels(n,1:length(next)) = next;    %  append to the labels matrix
   end
end

%  Read in the x-y data.  Use the vetorized fscanf function to load all
%  numerical values into one vector.  Then reshape this vector into a
%  matrix before copying it into the x and y matrices for return.

data = fscanf(fin,'%f');  %  Load the numerical values into one long vector

nd = length(data);        %  total number of data points
nr = nd/ncols;            %  number of rows; check (next statement) to make sure
if nr ~= round(nd/ncols)
   fprintf(1,'\ndata: nrow = %f\tncol = %d\n',nr,ncols);
   fprintf(1,'number of data points = %d does not equal nrow*ncol\n',nd);
   error('data is not rectangular')
end

data = reshape(data,ncols,nr)';   %  notice the transpose operator
x = data(:,1);
y = data(:,2:ncols);

%  end of readColData.m
```

VITA

Uchenna O. Odi was born in Lagos, Nigeria. He attended high school in Bartlesville, OK at Bartlesville High School. He graduated from high school in 2001 and started that same year attending the University of Oklahoma studying chemical engineering. During his time at the University of Oklahoma, he worked as a lab assistant for the chemical engineering and environmental engineering laboratories. He received his B.S. from the Univeristy of Oklahoma in 2006 and started working with Schlumberger in the same year. In 2007 he started attending Texas A&M University for a M.S. in petroleum engineering and received his M.S. in 2009. For contact information Uchenna can be reached at ouodi7@gmail.com. His permanent address is 5227 Maple Hill Trail Kingwood, TX 77345.